UNIVERSIDAD DE CUENCA
Facultad de Ingeniería
Carrera de Electrónica y Telecomunicaciones

Diseño e implementación de los sistemas de instrumentación, comunicaciones y control de un sistema multi-tanque

Trabajo de titulación previo a la obtención del título de Ingeniero en Electrónica y Telecomunicaciones

Autores :

Ricardo Sebastián Enderica Bustos	C.I. 0105750426
Fabricio Esteban López Alvarado	C.I. 0105879076

Director :

Ing. Andrés Marcelo Vázquez Rodas, PhD	C.I. 0301496840

Co-Director :

Ing. Luis Ismael Minchala Ávila, PhD	C.I. 0301453486

Cuenca - Ecuador

2018

Resumen

El advenimiento del nuevo paradigma de la cuarta revolución industrial, denominada Industria 4.0, busca la interconexión entre las diferentes plantas productivos con la finalidad de mejorar su eficiencia. Un requisito imprescindible de esta evolución es la interoperabilidad entre tecnologías de diferentes proveedores y fabricantes para coexistir dentro de los procesos industriales de manera eficiente. La variedad de soluciones propietarias existentes en el mercado provoca que la interoperabilidad entre sistemas y dispositivos se convierta en un reto, debido a la heterogeneidad producida por tal cantidad de tecnología que pertenece a diferentes proveedores, además de un alto capital que se requiere para adquirir estos sistemas y dispositivos. Por tanto, las pequeñas y medianas empresas industriales requieren buscar soluciones de bajo costo que cubran la interoperabilidad e interconexión entre los dispositivos y las plantas de producción.

Este trabajo propone el diseño e implementación de los sistemas de instrumentación, comunicaciones industriales y control de un sistema de dos tanques de agua mediante controladores lógicos programables (PLCs) de bajo costo y compatibles con un servidor de plataforma abierta para comunicaciones industriales de arquitectura unificada (OPC-UA) de software libre. El servidor OPC-UA permite la integración de la planta con un sistema de supervisión, control y adquisición de datos (SCADA). El propósito es explorar plataformas embebidas de bajo costo como *PiXtend* y *M-Duino* para generar funciones de comunicación, resguardando la interoperabilidad entre los dispositivos, preprocesamiento de variables y esquemas de control dentro de un ambiente industrial. La arquitectura del sistema de instrumentación y comunicación propuesto en este trabajo está basado en una red de bus de campo implementando el protocolo *Modbus*. Los sistemas de control propuestos se basan en controladores PID y de ganancia programada difusa (FGS-PID), con detección y diagnóstico de fallos mediante el filtro de Kalman extendido (EKF) usado para la estimación de los estados del sistema.

Palabras clave : Sistema multi-tanque, Detección y diagnóstico de fallas, Protocolo Modbus, Controlador PID, Controlador FGS-PID, Observador EKF, OPC-UA, SCADA.

Abstract

The advent of the new paradigm of the fourth industrial revolution, called Industry 4.0, seeks the interconnection between different plants and productive sectors in order to improve their efficiency. An essential requirement of this evolution is the interoperability between technologies from different suppliers and manufacturers to coexist efficiently within industrial processes. The variety of proprietary solutions existing in the market turns the interoperability between systems and devices a challenge, due to the heterogeneity produced by such amount of technology belonging to different suppliers, in addition to the high investment required to acquire these systems and devices. Therefore, small and medium-sized industrial companies need to look for low-cost solutions that cover the interoperability and interconnection between the devices and the production plants.

This work proposes the design and implementation of instrumentation, industrial communications and control systems of a two-tank water system through low-cost programmable logic controllers (PLCs) and compatible with an Open Platform Communications Unified Architecture (OPC-UA) open source server. The OPC-UA server allows the integration of the plant with a Supervisory Control and Data Acquisition (SCADA) system. The purpose is to explore embedded low-cost platforms such as *PiXtend* and *M-Duino* to generate communication functions, safeguarding interoperability between devices, preprocessing of variables and control schemes within an industrial environment. The architecture of the instrumentation and communication system proposed in this paper is based on a fieldbus network implementing the *Modbus* protocol. The proposed control schemes are based on Proportional Integral Derivative controllers (PID controllers) and Fuzzy Gain Scheduling controller (FGS-PID), with detection and diagnosis of failures capabilities by using Extended Kalman Filter (EKF) to estimate the states of the system.

Keywords : Multi-tank system, Detection and diagnosis of failures, Modbus protocol, PID controller, FGS-PID controller, EKF observer, OPC-UA, SCADA

Índice general

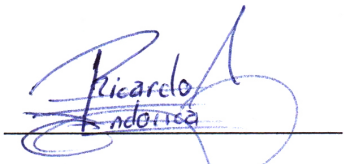
Índice de figuras

Índice de tablas

Cláusula de Propiedad Intelectual

Ricardo Sebastián Enderica Bustos, autor del trabajo de titulación "Diseño e implementación de los sistemas de instrumentación, comunicaciones y control de un sistema multi-tanque", certifico que todas las ideas, opiniones y contenidos expuestos en la presente investigación son de exclusiva responsabilidad de su autor.

Cuenca, 24 de octubre de 2018



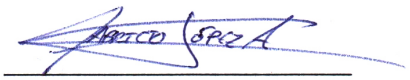
Ricardo Sebastián Enderica Bustos

C.I: 010575042-6

Cláusula de Propiedad Intelectual

Fabricio Esteban López Alvarado, autor del trabajo de titulación "Diseño e implementación de los sistemas de instrumentación, comunicaciones y control de un sistema multi-tanque", certifico que todas las ideas, opiniones y contenidos expuestos en la presente investigación son de exclusiva responsabilidad de su autor.

Cuenca, 24 de octubre de 2018



Fabricio Esteban López Alvarado

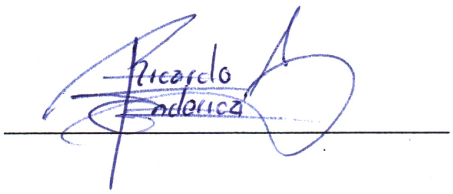
C.I: 010587907-6

Cláusula de licencia y autorización para publicación en el Repositorio Institucional

Ricardo Sebastián Enderica Bustos en calidad de autor y titular de los derechos morales y patrimoniales del trabajo de titulación "Diseño e implementación de los sistemas de instrumentación, comunicaciones y control de un sistema multi-tanque", de conformidad con el Art. 114 del CÓDIGO ORGÁNICO DE LA ECONOMÍA SOCIAL DE LOS CONOCIMIENTOS, CREATIVIDAD E INNOVACIÓN reconozco a favor de la Universidad de Cuenca una licencia gratuita, intransferible y no exclusiva para el uso no comercial de la obra, con fines estrictamente académicos.

Asimismo, autorizo a la Universidad de Cuenca para que realice la publicación de este trabajo de titulación en el repositorio institucional, de conformidad a lo dispuesto en el Art. 144 de la Ley Orgánica de Educación Superior.

Cuenca, 24 de octubre de 2018



Ricardo Sebastián Enderica Bustos

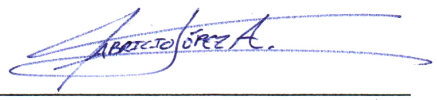
C.I: 010575042-6

Cláusula de licencia y autorización para publicación en el Repositorio Institucional

Fabricio Esteban López Alvarado en calidad de autor y titular de los derechos morales y patrimoniales del trabajo de titulación "Diseño e implementación de los sistemas de instrumentación, comunicaciones y control de un sistema multi-tanque", de conformidad con el Art. 114 del CÓDIGO ORGÁNICO DE LA ECONOMÍA SOCIAL DE LOS CONOCIMIENTOS, CREATIVIDAD E INNOVACIÓN reconozco a favor de la Universidad de Cuenca una licencia gratuita, intransferible y no exclusiva para el uso no comercial de la obra, con fines estrictamente académicos.

Asimismo, autorizo a la Universidad de Cuenca para que realice la publicación de este trabajo de titulación en el repositorio institucional, de conformidad a lo dispuesto en el Art. 144 de la Ley Orgánica de Educación Superior.

Cuenca, 24 de octubre de 2018



Fabricio Esteban López Alvarado

C.I: 010587907-6

Dedicatoria

A mi madre Jenny:

Por su amor y valentía, por cuidar de mi y apoyarme en todo momento. Porque gracias a ella he podido superar varias etapas de mi vida. Gracias por estar siempre velando por mi, te amo mamá.

A mis abuelos:

Por brindarme su apoyo y cariño. Porque ustedes siempre me enseñaron buenos valores y por ser junto a mi madre el pilar de mi vida.

A mi familia:

Por estar dispuestos a brindarme su ayuda incondicional en cualquier momento. Porque sin su apoyo nunca hubiera alcanzado ningún logro.

Sebastián Enderica B.

Dedicatoria

A mis padres:

Quienes me dieron la luz de la existencia, especialmente a mi madre Bertha por su dedicación y el amor brindado, y que desde su morada celestial estará compartiendo conmigo este triunfo.

A mi hermana Maritza:

Por todo su cariño y por ser mi apoyo permanente y que juntos hemos logrado superar todas las dificultades que se han presentado en nuestras vidas.

A mi familia materna:

Que con mucho amor y esfuerzo supieron abrirnos las puertas de su hogar y de su corazón y que fueron los pilares fundamentales para la culminación de mi carrera universitaria.

A mi abuelito y mi tía:

Quienes mientras vivían, estuvieron apoyándome y respaldándome en toda mi formación.

A todos ustedes que supieron comprenderme y me ayudaron a superar todos los momentos difíciles de mi vida.

Fabricio López A.

Agradecimientos

A mi familia, por todo el apoyo que me han entregado durante mi formación como estudiante.

A los todos los profesores que han sido parte de mi formación académica en la Universidad. Especialmente a nuestros tutores, los ingenieros Andrés Vázquez e Ismael Minchala, por brindarnos sus conocimientos y tiempo en este trabajo. A los ingenieros Daniel Merchán y Jonnathan Peralta por su motivación y respaldo en la culminación de este trabajo.

Finalmente, a mis compañeros y amigos, por hacer de la vida universitaria una etapa entretenida, En especial a Fabricio, por su responsabilidad y dedicación en el desarrollo de este trabajo.

Sebastián Enderica B.

Agradecimientos

Quiero agradecer a toda mi familia por ser el pilar fundamental durante mi formación universitaria.

A los ingenieros Andrés Vázquez e Ismael Minchala como tutores del proyecto por brindarnos todos sus conocimientos y dedicarnos su tiempo. A los ingenieros Daniel Merchán y Jonnathan Peralta por su motivación y respaldo en la consecución de este trabajo.

Finalmente agradezco a mis compañeros y amigos que han sido un gran apoyo, en especial a Sebastián por su responsabilidad y esfuerzo para superar las dificultades presentadas en el desarrollo de este trabajo.

Fabricio López A.

Abreviaciones y Acrónimos

ADU Aplication Data Unit. 21
AMQP Advanced Message Queuing Protocol. 14
API Application Programming Interface. 14, 15, 33

CAN Control Area Network. 20, 34
CIM Computer Integrated Manufacturing. 10, 11, 34, 47
CIP Control Industrial Protocol. 21
COM Component Object Model. 16
CPS Cyber-Physical System. 2, 3, 35
CPU Central Processing Unit. 51
CRM Customer Relationship Management. 8
CSP Constraint Satisfaction Problem. 33

DCOM Distributed Component Object Model. 16, 21
DIUC Dirección de Investigación de la Universidad de Cuenca. 4

EKF Extended Kalman Filter. 29, 38, 60, 63, 66, 68, 73, 74, 79, 90, 94
ERP Enterprise Resource Planning. 8, 11
EWMA Exponentially Weighted Moving Average. 54

FGS Fuzzy Gain Scheduling. 4, 25, 38, 49, 50, 60, 61, 67, 68, 70, 72–76, 78, 79, 82, 86–90, 94

GPIO General Purpose Input/Output. 44

HMI Human Machine Interface. 4, 32, 44, 45, 48–51, 53

IIC Industrial Internet Committee. 6, 9
IIoT Industrial Internet of Things. 6, 8–10, 16, 32, 33, 38, 47
IoT *Internet de las Cosas*. 6, 18, 19
IP Internet Protocol. 35

KF Kalman Filter. 28

LAN Local Area Network. 32
LDAP Lightweight Directory Access Protocol. 35

M2M Machine to Machine. 6
MES Manufacturing Execution Systems. 11
MQTT Message Queuing Telemetry Transport. 14
MRP Manufacturing Resources Planning. 11
MTU Master Terminal Unit. 17, 18, 32

NFV Network Functionality Virtualization. 2

OPC OLE for Process Control. 3, 12, 13, 16, 33, 34
OPC A&E OLE for Process Control Alarms and Events. 13, 16
OPC DA OLE for Process Control Data Access. 12, 13, 16
OPC HDA OLE for Process Control Historical Data Access. 13, 16
OPC XML-DA OLE for Process Control XML Data Access. 35
OPC-UA Open Platform Communications Unified Architecture. 3, 13–16, 32–35, 38, 48, 49, 69, 90, 92, 94, 95
OSACIM Open Source Architecture for Advanced Computer Integrated Manufacturing. 34, 35
OSI Open System Interconnection. 19, 21

P&ID Process and Instrumentation Diagram. 4, 39, 44, 58, 94
PDU Protocol Data Unit. 21, 22
PID Proportional Integral Derivative Controller. 4, 24, 25, 38, 49, 50, 60, 61, 67, 68, 70, 72–90, 94
PLC Programmable Logic Controller. 3, 4, 11, 18, 38, 42, 43, 45, 47–49, 91, 94, 95, 99, 118
PWM Pulse-Width Modulation. 45

RPC Remote Procedure Call. 21
RTU Remote Terminal Unit. 17, 18, 32

SCADA Supervisory Control And Data Acquisition. 3, 11, 12, 17–19, 32, 34, 95
SDN Software Defined Network. 2, 32, 35
SNMP Simple Network Management Protocol. 35
SOA Service Oriented Architecture. 16
SQL Structured Query Language. 35

TCP/IP Transmission Control Protocol over Internet Protocol. 19, 21

UDP User Datagram Protocol. 14

WSM Warehouse Stock Management. 8

Capítulo 1

Introducción

Este capítulo presenta los antecedentes, el problema de investigación, los objetivos del trabajo y las contribuciones de la tesis.

1.1. Antecedentes

El desarrollo industrial en Latinoamérica ha ganado una gran importancia en la última década debido a que la industrialización es considerada como el distintivo entre los países desarrollados y subdesarrollados. Ecuador también busca invertir en el desarrollo industrial con el fin de mejorar su producción y por tanto incrementar la participación de la industria en la economía del país [12]. El principal obstáculo para que los países subdesarrollados se industrialicen es el alto capital necesario para la actualización de tecnologías [13]. El desarrollo de una industria implica el crecimiento de la demanda de un producto o la elaboración de nuevos productos. Esto provoca la creación de nuevos sectores productivos que demandan actualización tecnológica y la adquisición de nuevos bienes. El crecimiento no planificado en la industria de manufactura y transformación primaria, suele generar heterogeneidad de tecnologías y edad tecnológica de las maquinarias [14].

El advenimiento del nuevo paradigma de la cuarta revolución industrial, denominada Industria 4.0, busca la interconexión entre las diferentes plantas y sectores de producción con el fin de mejorar la eficiencia de la utilización de recursos y la flexibilidad de producción [15]. Además sirve como medio para la implementación de tecnologías como procesamiento y almacenamiento en la nube o en la niebla, redes definidas por software (SDN por sus siglas en inglés), virtualización de funciones de red (NFV, por sus siglas en inglés) de modo que se pueda manejar gran cantidad de datos que puedan ser analizados con el fin de mejorar la autonomía de los procesos físicos de producción, monitorización y mantenimiento de las plantas dando paso al paradigma de los sistemas ciber físicos (CPS, por sus siglas en inglés) [1]. Un requisito imprescindible de esta evolución es la interoperabilidad entre tecnologías de diferentes proveedores y fabricantes para coexistir dentro de los procesos industriales de manera eficiente [16].

Las opciones tecnológicas disponibles en el mercado industrial han sido mayormente desarrolladas a lo largo de los años por fabricantes de tecnologías propietarias. La variedad de soluciones propietarias existentes en el mercado provoca que la interoperabilidad entre sistemas y dispositivos se convierta en un reto, debido a la heterogeneidad producida por tal cantidad de tecnología que pertenece a diferentes proveedores. Los problemas causados por la interoperabilidad entre las tecnologías de distintos fabricantes generan dependencia a las soluciones y productos que ofrecen ciertos fabricantes. Por lo tanto, hasta que exista una estandarización de las interfaces, los protocolos y las aplicaciones, la interconexión requerida para la implementación de la industria 4.0 sera una opción potencialmente costosa, ineficiente y posiblemente insegura [1].

En resumen, Latinoamérica tiene un gran interés de desarrollar el sector industrial, para ello las industrias deben estar a la vanguardia con el paradigma de la Industria 4.0. Por tanto, las industrias requieren de interoperabilidad e interconexión entre las plantas de producción y compatibilidad con las nuevas tecnologías de las redes de computadoras. Esto conlleva a

que se busquen dispositivos inteligentes y soluciones de bajo costo que sean accesibles para las pequeñas y medianas empresas.

Para satisfacer el requisito de interoperabilidad entre tecnologías de distintos fabricantes, es necesaria la implementación de alguna solución estándar que permita el intercambio de información en procesos industriales. El estándar OPC-UA (Arquitectura Unificada) es una de las soluciones de este tipo [16], que en conjunto con la implementación de un sistema SCADA permite brindar la capacidad de comunicación, supervisión y control entre los diferentes dispositivos que conforman un proceso de automatización industrial [17].

En este contexto, este trabajo propone el diseño e implementación de los sistemas de instrumentación, comunicaciones industriales y control de un sistema de dos tanques de agua mediante PLCs de bajo costo y compatibles con un servidor OPC desarrollado en software libre. El propósito es explorar plataformas embebidas de bajo costo como *PiXtend*[18] y *M-Duino*[19] para generar funciones de comunicación, pre-procesamiento de variables y esquemas de control dentro de un ambiente industrial, de manera que pueda integrarse de manera abierta a soluciones del paradigma de la industria 4.0 y el paradigma de los sistemas CPS a través del servidor OPC-UA.

1.2. Identificación del problema

El mercado industrial ha sido acaparado mayormente por fabricantes de tecnologías propietarias a lo largo de los años. Esto causa que la interoperabilidad entre sistemas y dispositivos se convierta en un reto, debido a la heterogeneidad producida por tal cantidad de tecnología que pertenece a diferentes proveedores; de igual manera, es posible que esto genere dependencia a las soluciones que ofrecen ciertos fabricantes y sus tecnologías propietarias. En ocasiones, este tipo de equipamiento resulta costoso debido a que implica esquemas de licenciamiento de software de desarrollo y configuraciones específicas/modulares de hardware.

Para solventar los problemas mencionados, resulta importante la búsqueda de soluciones de hardware y software libre, que potencien desarrollos complejos a nivel de supervisión, control industrial y la interoperabilidad entre plantas de producción, pero con precios asequibles para empresas de pequeña y mediana envergadura.

Entre las opciones de hardware libre para aplicaciones de control y automatización industrial, de arquitectura libre, se encuentran controladores lógicos programables (PLC, por sus siglas en inglés) embebidos de bajo costo como el *PiXtend* basado en la plataforma *Raspberry Pi* y *M-Duino* basado en la plataforma *Arduino*. Bajo esta filosofía, se evaluarán también la comunicación entre la planta y un servidor OPC de software libre desarrollado en *Python*.

Específicamente, este proyecto propone el diseño e implementación de los sistemas de instrumentación, comunicaciones y control de un sistema hidráulico multi-tanque mediante PLCs embebidos de bajo costo. La interoperabilidad entre dispositivos en la red industrial se garantiza con aplicaciones desarrolladas en hardware y software libre. Este trabajo forma parte de las actividades del proyecto DIUC: *"Prototipo de expansión de funcionalidades para dispositivos de automatización industrial clásicos utilizando plataformas embebidas de bajo costo"*, el cual se encuentra en desarrollo por parte del Departamento de Ingeniería Eléctrica, Electrónica y Telecomunicaciones de la Facultad de Ingeniería de la Universidad de Cuenca.

1.3. Objetivos

1.3.1. Objetivo general

Diseñar e implementar los sistemas de instrumentación, comunicaciones industriales y de control de un sistema de dos tanques de agua.

1.3.2. Objetivos específicos

- Diseñar el P&ID del proceso de dos tanques.
- Modelizar el sistema de dos tanques usando ecuaciones de continuidad y ecuaciones diferenciales.
- Calibrar los instrumentos del proceso: válvulas de control, sensores de flujo, nivel.
- Implementar el esquema de control FGS-PID en el *PiXtend*.
- Desarrollar una interfaz HMI con información de variables de proceso y control.
- Realizar el montaje de elementos de medición y control en un tablero.

1.4. Contribuciones de la tesis

Este trabajo presenta las siguientes contribuciones:

- Un sistema multi-tanque que consta de dos tanques de reserva, un tanque de bombeo y su sistema de instrumentación integrado en un tablero de control.
- Un sistema de control y detección de fallas mediante el uso de software y hardware libre.
- Un sistema de comunicación industrial basado en software libre que permite la interoperabilidad de equipos de diferentes fabricantes.

Capítulo 2

Fundamentos teóricos

En el presente capítulo, se expone una visión general de los principales conceptos relacionados con los sistemas de comunicación y control de un proceso industrial automatizado.

2.1. Industria 4.0

La arquitectura de la Industria 4.0, conocida como internet industrial de las cosas (IIoT, por sus siglas en inglés) se considera como la evolución de las redes industriales maquina a máquina (M2M, por sus siglas en inglés). La diferencia fundamental diferencia entre estos paradigmas es la adición de Internet como se muestra en la Figura 2.1.

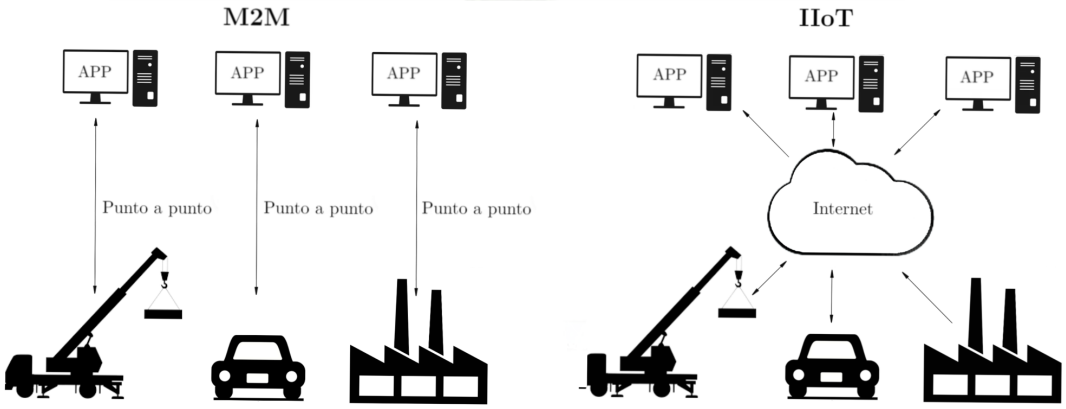


Figura 2.1: Comparación del paradigma M2M a IIoT [1]

El IIoT es un subconjunto del Internet de las cosas (IoT, por sus siglas en inglés). Si bien el IoT abarca la interconexión de todo como consumidor, industria, empresa, negocios; el IIoT se enfoca en la energía, salud, manufactura, sector público, transporte y los sistemas industriales. El IIoT busca interconectar sistemas industriales a internet e integrar estos sistemas con sistemas empresariales con el fin de mejorar los procesos de producción y los procesos de negocio. El paradigma del IIoT consiste en que los sistemas industriales integrados a internet proporcionan datos operacionales a los sistemas de almacenamiento de la empresa para realizar preprocesarlos y realizar un análisis histórico y predictivo avanzado en la nube o en la niebla. Estos análisis impulsaran la toma de decisiones de manera óptima, la operación eficiente y autonomía de sistemas de control industrial. Para cubrir estos objetivos es necesaria una arquitectura abierta, aplicable y escalable [1].

2.1.1. Arquitectura de referencia del IIoT

El consorcio del internet industrial (IIC, por sus siglas en inglés) ha proporcionado un modelo arquitectónico de referencia que se puede implementar como bloques de construcción interope-

rables e intercambiables. Esta arquitectura de referencia está diseñada para ser flexible y cubrir una amplia gama de escenarios de implementación en muchas industrias, como la energía, la atención médica y el transporte [1].

El modelo de referencia se presenta como un modelo flexible y sin un conjunto mínimo de dominios funcionales requeridos para su aplicación. De manera general los cinco dominios fundamentales del modelo de referencia:

- Dominio de control
- Dominio de operaciones
- Dominio de información
- Dominio de la aplicación
- Dominio comercial

2.1.1.1. Dominio de control

El dominio de control es típicamente realiza tareas tales como leer datos de sensores, luego las unidades lógicas aplican reglas, lógica y posteriormente aplican retroalimentación a las máquinas para ejercer control sobre el proceso. Esto podría requerir funciones como:

- Funciones de comunicación: conectan todos los sensores funcionales, controladores de actuadores, dispositivos de E/S remotas y, en última instancia, *gateways* de enlace a un protocolo común.
- Funciones de modelado de datos: realizan un análisis sobre la representación de los estados, las condiciones y los comportamientos del sistema bajo control.
- Funciones de gestión de activos: permite que los sistemas de control se incorporen, configuren, establezcan políticas y realicen autodiagnósticos y actualizaciones automáticas de firmware y software.
- Funciones de ejecución: son realizadas por el operador final, el operador realiza funciones de supervisión del entorno y el comportamiento del sistema bajo control.

2.1.1.2. Dominio operacional

El dominio de operaciones consta de bloques de pronóstico, optimización, monitoreo y diagnóstico, aprovisionamiento e implementación, y administración del sistema. Estas funciones se relacionan directamente con las funciones de control inferiores y, en última instancia, con el usuario.

- El bloque de administración de sistemas ofrece un conjunto de herramientas o funciones que permiten a las funciones de administración de activos tener comunicaciones bidireccionales con otros activos (dispositivos, actuadores y sensores) para controlar y administrar activos remotos.
- El bloque de Monitoreo y Diagnóstico es un valioso conjunto de funciones que permiten la detección y predicción de las ocurrencias de fallas o problemas.
- El bloque de pronóstico se deriva del conjunto de funciones que brindan monitorización y diagnóstico, ya que sirven como motor de análisis predictivo.
- El bloque de optimización es un conjunto de funciones destinadas a mejorar el rendimiento y la eficiencia de los activos.

2.1.1.3. Dominio de información

El Dominio de información tiene el objetivo de transformar los datos brutos, recolectados de los otros dominios, en información, que posteriormente puede utilizarse como control de retroalimentación para estabilizar o mejorar el proceso. También consta de funciones de procesamiento de la calidad de los datos, como el filtrado de datos, la limpieza y la eliminación de datos duplicados. Además, hay funciones que permiten la transformación sintáctica y semántica, que se relacionan con obtener el formato y el significado del mensaje correcto.

2.1.1.4. Dominio de aplicación

El dominio de la aplicación es responsable de las funciones que controlan la lógica y brindan funcionalidades comerciales específicas. Las aplicaciones no controlan los procesos de la máquina o del sistema, ya que esa es la función del dominio de control. El dominio de la aplicación admite interfaces de usuario que permiten la interacción con la aplicación.

2.1.1.5. Dominio empresarial

El dominio empresarial permite la integración y la compatibilidad entre las funciones del IIoT y los sistemas empresariales, como gestión de recursos empresariales (ERP, por sus siglas en inglés), gestión de relaciones con los clientes (CRM, por sus siglas en inglés), gestión de existencias de almacén (WSM, por sus siglas en inglés) y muchos otros.

2.1.2. Topología arquitectónica

La topología arquitectónica propuesta por el IIC tiene un diseño de tres niveles. También tiene convertidores de medios de entrada en las áreas fronterizas para convertir y traducir protocolos y tecnologías heterogéneas. La arquitectura también admite conectividad directa a la niebla y una topología de análisis distribuido en la nube. En la Figura 2.2 se presenta la relación entre los dominios funcionales y la topología de la arquitectura de tres niveles. Sin embargo, no hay exclusividad entre un dominio funcional y un nivel, ya que pueden solicitar servicios entre ellos [1].

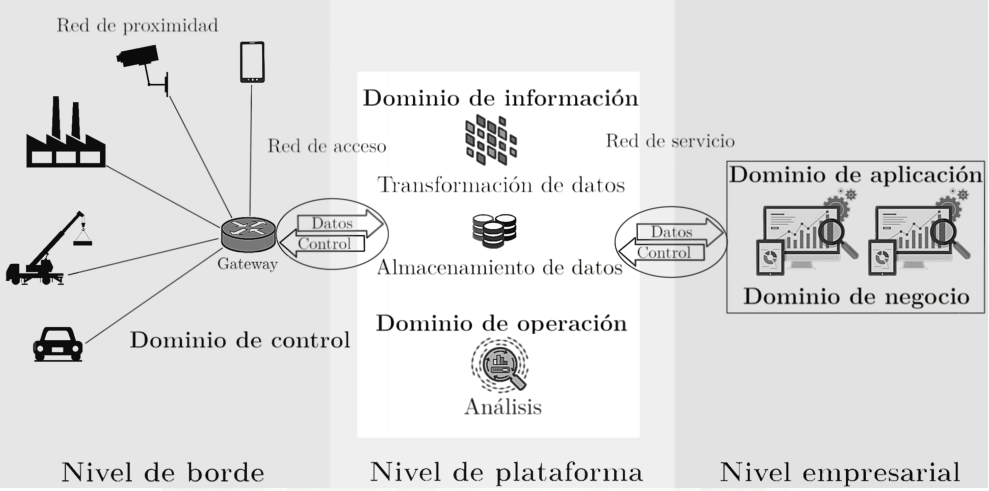


Figura 2.2: Relación entre los dominios funcionales y los niveles de la arquitectura del IIoT [1]

- Nivel del borde: nivel donde los datos de todos los nodos finales se recopilan, agregan y transmiten a través de la red de proximidad a un *gateway* fronterizo. Dependiendo de los protocolos y las tecnologías que se utilicen dentro del nivel de borde, se puede aplicar alguna traducción de datos e integración de interfaz en concentradores o conversores de protocolos. El nivel de borde contiene las funciones para el dominio de control.
- Nivel de plataforma: El nivel de plataforma recibe datos del nivel de borde sobre la red de acceso y es responsable de la transformación y procesamiento de datos. El nivel de la plataforma también es responsable de administrar los datos de control que fluyen desde la empresa hasta los niveles del borde. La mayoría de las funciones relacionadas con los dominios de información y operaciones están dentro del nivel de plataforma.
- Nivel empresarial: El nivel empresarial implementa la lógica de aplicaciones y negocios para los sistemas de soporte de decisiones y las interfaces para especialistas en operaciones. En este nivel están la mayoría de las funciones de dominio de aplicaciones y negocios.

2.1.3. Características clave del IIoT

Algunas de las características clave de la función de conectividad son el rendimiento, determinado por baja latencia y fluctuación, alto rendimiento, elasticidad y escalabilidad. En IIoT, la latencia y la inestabilidad son posiblemente los factores más importantes ya que el IIoT generalmente requiere tiempos de reacción muy cortos ya que sus acciones a menudo se coordinan estrechamente con los procesos del mundo real. La escalabilidad también es un factor muy deseable, ya que en algunos casos de uso del IIoT se pueden conectar millones de sensores finales para que el IIoT pueda crecer y acomodarlos a todos.

La red debe ser lo suficientemente robusta como para sobrevivir a las duras condiciones del mundo real. En este tipo de entorno, el punto extremo fallará, por lo que será necesario localizar la pérdida y el alcance de las comunicaciones [1].

2.2. Manufactura integrada por computador

La manufactura integrada por computador (CIM, por sus siglas en inglés) describe el concepto de mejorar el rendimiento industrial mediante el uso de computadoras y técnicas de automatización, las cuales se enfocan en optimizar las diferentes etapas de una línea de producción [20]. CIM involucra el proceso de producción completo, desde el proveedor hasta el consumidor final [21]. En este contexto, CIM plantea la integración computarizada de todos los aspectos de la industria, incluyendo subprocesos como el diseño de productos, proceso de fabricación, control de planta, automatización de procesos y soporte comercial [22].

2.2.1. Pirámide CIM

La arquitectura CIM se representa en 5 niveles, formando la denominada pirámide CIM, como se muestra en la Figura 2.3. La pirámide integra los diferentes procesos involucrados en la producción, desde lo correspondiente a la manufactura del producto hasta el nivel de gestión empresarial. Para garantizar la integración de los niveles de la pirámide se debe establecer redes de comunicación industriales que permitan el intercambio de información entre los diferentes niveles de la pirámide [23].

El nivel 0 corresponde al nivel de campo, en el cual se encuentran los sensores y actuadores que funcionan como interfaz con las variables físicas del proceso, recolectando datos y ejecutando acciones de control. El nivel 1 corresponde al nivel de proceso, el cual contiene los dispositivos encargados del análisis y procesamiento de datos, en este nivel se encuentran elementos como

los PLCs. El nivel 2 corresponde al nivel de supervisión, en este punto se ubica el SCADA y se encarga de recibir información del proceso con el propósito de realizar supervisión y control de la planta, además de actuar como interfaz para los operadores [2]. El nivel 3 corresponde al nivel de gestión, en la cuál se realiza la planificación de producción y manufactura, en este nivel se encuentran conceptos como los sistemas de ejecución de manufactura (MES, por sus siglas en inglés) y planificación de recursos de manufactura (MRP, por sus siglas en inglés). El nivel 4 corresponde al nivel de empresa que se encarga de la logística general de la planta productiva [23], incluye el concepto de sistemas de planificación de recursos empresariales (ERP, por sus siglas en inglés), el cual ayuda en la organización de la información del negocio mediante una base de datos que incluye gestión de inventario, gestión de pedidos de clientes, planificación y gestión de producción, contabilidad, gestión de recursos humanos [24].

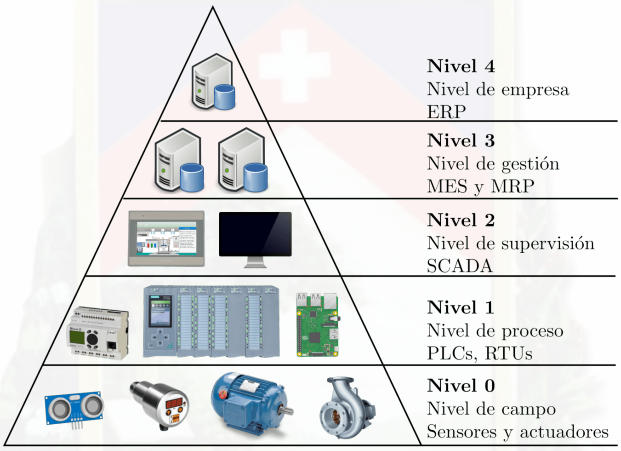


Figura 2.3: Pirámide CIM [2]

2.2.2. Beneficios de CIM

La implementación de CIM en un proceso de producción mejora el rendimiento general del mismo. CIM ayuda a las empresas con el manejo de información, permitiendo organizar y coordinar los diferentes tipos de datos generados en el proceso, como datos de funcionamiento, de producción, operacionales y de desempeño. Otro beneficio que presenta CIM es el ahorro de costos asociados con el papel, debido a que diferentes procesos de ingeniería y planificación generan reportes que deben ser distribuidos a diferentes departamentos en la planta. Tradicionalmente esto se realizaba mediante documentos impresos, con CIM esta información se puede distribuir por medio de los sistemas de comunicación, permitiendo su visualización en terminales. Otros beneficios de los sistemas de comunicación implementados en CIM son que el intercambio de información es realizado de forma más rápida y se pueden sincronizar procesos para ejecutar tareas de manera paralela, reduciendo los tiempos de producción [22].

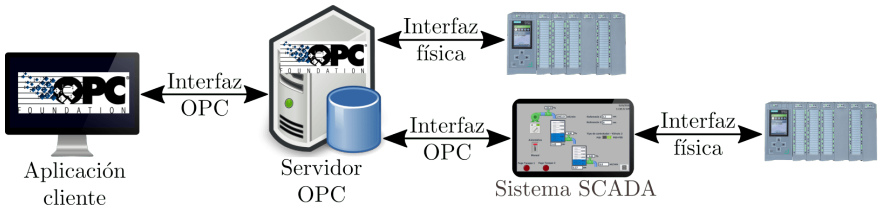


Figura 2.4: Interacción del estándar OPC con un sistema industrial [3]

2.3. Estándar OPC

En un ambiente industrial, el uso de software para el acceso a los datos generados por los diferentes dispositivos requiere *drivers* para la interacción con el software de recolección de datos. Cada fabricante provee *drivers* específicos para cada dispositivo, esto implica que cada aplicación de software debe incluir los *drivers* de todos los dispositivos que se requieren manejar. Esto generalmente implica que se pueden presentar problemas de compatibilidad entre *drivers* de diferentes fabricantes [25].

El estándar OPC se desarrolló como una solución para permitir que aplicaciones cliente accedan a los datos proporcionados por dispositivos. OPC actúa como una interfaz estandarizada para solventar problemas de compatibilidad con los *drivers* de los diferentes dispositivos [3]. En la Figura 2.4 se muestra la interacción del estándar OPC con otros componentes del sistema industrial. El servidor OPC es el elemento principal dentro de esta arquitectura, ya que: establece interfaces físicas con los dispositivos de entrada y salida, presenta una interfaz de comunicación con el sistema SCADA y permite el intercambio de toda esta información con aplicaciones cliente mediante una interfaz OPC.

2.3.1. Especificaciones del estándar OPC

Las especificaciones más relevantes del estándar OPC definen interfaces comunes para el intercambio de datos, eventos e información entre dispositivos y aplicaciones, basada en una arquitectura cliente-servidor [26].

2.3.1.1. Acceso de datos (OPC DA)

Define una interfaz para el intercambio de datos entre un cliente y servidor mediante un flujo constante de datos. El servidor permite la conexión de uno o varios clientes, mientras que un cliente puede estar conectado a varios servidores [26].

2.3.1.2. Alarmas y eventos (OPC A&E)

Define una interfaz entre el servidor y los clientes para la gestión estructurada de alarmas y eventos. El servidor puede recibir información de diferentes fuentes, analizar los datos y determinar si ocurrió un evento, o en su defecto puede recibir directamente las alarmas y eventos generados por otros dispositivos. El servidor envía a los clientes información sobre alarmas y eventos que han ocurrido [26].

2.3.1.3. Acceso de datos históricos (OPC HDA)

El servidor OPC HDA ofrece datos de proceso históricos a los clientes, además de los datos de proceso, el cliente puede solicitar la creación de datos procesados, denominados *datos agregados* [26].

Se distinguen dos tipos de implementaciones OPC HDA:

1. La tarea principal de servidor es almacenar datos para brindar tendencias simples.
2. El servidor realiza tareas como compresión y análisis de datos.

2.3.1.4. OPC por lotes

Es una extensión de la especificación OPC DA para el caso de procesos por lotes. Esta especificación ofrece soluciones de interoperabilidad de los componentes del proceso por lotes en un ambiente heterogéneo [26].

2.3.2. Arquitectura unificada OPC (OPC-UA)

OPC-UA es un estándar abierto independiente de la plataforma que permite el intercambio de información de manera robusta y segura entre dispositivos pertenecientes a un sistema.

OPC-UA permite trabajar de dos maneras: mediante un modelo cliente-servidor o mediante un modelo publicación-suscripción (*PubSub*). En el primer caso se realiza un intercambio de mensajes de solicitud y respuesta entre el cliente y servidor. En el caso del modelo *PubSub* se utiliza un *middleware* orientado a mensajes. Los publicadores envían mensajes al *middleware* sin tener conocimiento de los suscriptores, mientras que los suscriptores solicitan los datos de su interés sin conocer que publicadores existen. El *middleware* orientado a mensajes puede ser implementado de dos maneras:

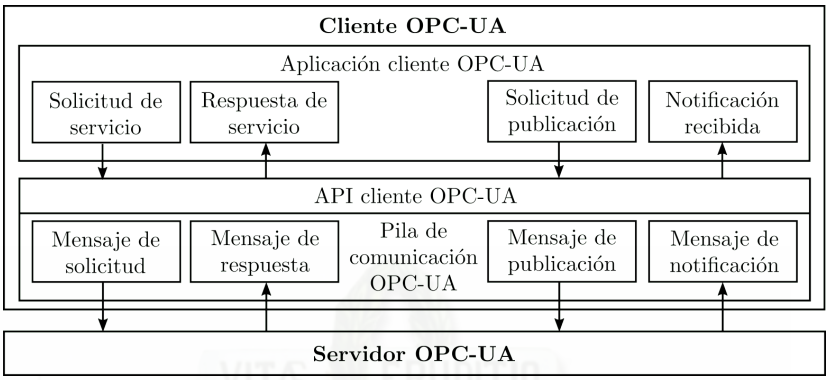


Figura 2.5: Arquitectura de un cliente OPC-UA [4]

- Sin uso de *broker*: el *middleware* orientado a mensajes es la infraestructura de red y se encarga de enrutar mensajes de tipo datagrama. Los publicadores y suscriptores utilizan protocolos de datagramas, por ejemplo UDP *multicast*.
- Basado en *broker*: el *middleware* orientado a objetos actúa como intermediario o *broker*. Los publicadores y suscriptores utilizan protocolos como AMQP [27] o MQTT [28] para comunicarse con el *broker*. Los mensajes se publican en colas específicas que el *broker* expone, las cuales pueden ser escuchadas por los suscriptores.

Cada servidor define el conjunto de servicios que soporta. Un servidor puede integrar datos actuales, alarmas, eventos y datos históricos dentro de un espacio de direcciones. Los clientes pueden acceder a la información de los servidores por medio del conjunto de servicios ofertados por el servidor [4].

2.3.2.1. Clientes OPC-UA

La arquitectura de un cliente OPC-UA se muestra en la Figura 2.5, donde se distinguen tres capas principales:

- Aplicación cliente: Define las funciones del cliente OPC-UA y utiliza llamadas de la API cliente para el intercambio de mensajes con el servidor.
- API cliente: La interfaz de programación de aplicaciones (API, por sus siglas en inglés) permite aislar el código de la aplicación de la pila de comunicación.
- Pila de comunicación OPC-UA: Se encarga del manejo de los mensajes intercambiados con el servidor y se comunica con la aplicación a través de la API cliente.

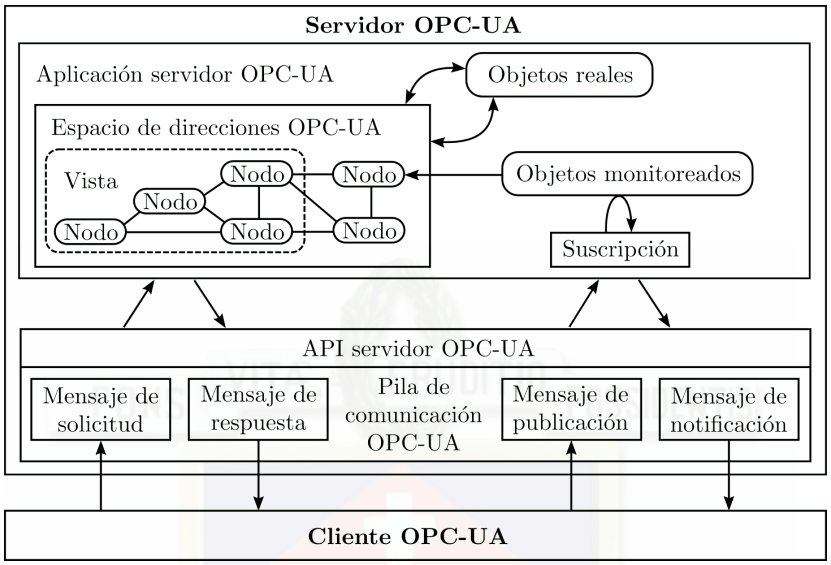


Figura 2.6: Arquitectura de un servidor OPC-UA [4]

2.3.2.2. Servidores OPC-UA

En la Figura 2.6 se muestra la arquitectura de un servidor OPC-UA, al igual que en el caso del cliente, se distinguen: la capa de aplicación, la pila de comunicación encargada del manejo de los mensajes y la API servidor, encargada de ser la interfaz entre las otras dos capas.

La aplicación servidor posee una estructura más compleja en comparación con la del cliente, debido a que implementa un mayor número de operaciones. En la Figura 2.6 se distinguen los siguientes elementos:

- Objetos reales: Son objetos físicos o de software a los cuales el servidor tiene acceso.
- Espacio de direcciones: Se modela como un conjunto de nodos, los cuales representan objetos reales accesibles a los clientes y contiene la información relacionada con los nodos y referencias entre dichos elementos. Mediante las referencias se pueden organizar los nodos ya sea en estructuras jerárquicas, mallas o cualquier combinación.
- Vistas: Son subconjuntos del espacio de direcciones, que permiten restringir la cantidad de información visible para un cliente.
- Elementos monitoreados: Son entidades dentro del servidor creadas por el cliente, permiten el monitoreo de un nodo, de manera que al detectarse un cambio en el mismo se genera una notificación enviada al cliente por medio de suscripción.
- Suscripciones: Permite la publicación de notificaciones a los clientes, los cuales pueden controlar la tasa a la cual se producen las publicaciones.

2.3.3. Conjuntos de servicios de OPC-UA

Los servicios soportados por un servidor se dividen en conjuntos de servicios, los conjuntos de servicios disponibles en OPC-UA son los siguientes [4]:

- **Descubrimiento**: Permiten que los clientes encuentren servidores disponibles en un sistema e informan de la configuración requerida para su conexión.
- **Canal seguro:** Permiten establecer un canal de comunicación que asegure la confidencialidad e integridad de los mensajes intercambiados con el servidor.
- **Sesión:** Establecen la conexión en capa de aplicación para la sesión de un usuario específico.
- **Manejo de nodos:** Permiten a los clientes la configuración del espacio de direcciones del servidor, permitiendo la agregación, eliminación o modificación de nodos en el servidor.
- **Vista:** Permiten al cliente descubrir los nodos que pertenecen a una vista, la vista por defecto es todo el espacio de direcciones.
- **Consulta:** Permiten a los clientes utilizar un criterio para seleccionar un grupo de nodos de interés dentro de una vista.
- **Atributo:** Permiten la lectura y escritura de los datos de los nodos almacenados en el servidor.
- **Método:** Los métodos son llamados de función de objetos. Cada objeto tiene sus propios métodos, lo cuales pueden ser descubiertos por los clientes.
- **Elemento monitoreado:** Permiten a los clientes la creación, manejo de elementos monitoreados y configurar la tasa de publicación de notificaciones.
- **Suscripción:** Permiten al cliente la gestión de suscripciones para el monitoreo de nodos específicos.

Considerando que el estándar OPC se encuentra basado en tecnología COM/DCOM de *Microsoft* [25], la principal ventaja de OPC-UA es que se encuentra basado en una arquitectura orientada a servicios (SOA, por sus siglas en inglés). El uso de SOA garantiza su interacción con una mayor cantidad de dispositivos debido a que es independiente de la plataforma [29], lo cual es compatible con los paradigmas del IIoT y la Industria 4.0 [30]. Adicionalmente OPC-UA permite unificar los clásicos servidores OPC DA, OPC A&E y OPC HDA por un único conjunto de servicios, ofreciendo una comunicación fácil, segura y eficiente [29].

2.4. Sistema SCADA

El sistema de supervisión, control y adquisición de datos (SCADA, por sus siglas en inglés) permite la centralización de un sistema naturalmente distribuido. De esta manera un operador y/o administrador puede monitorear y controlar elementos de un proceso desde una estación sin necesidad de desplazarse hacia el punto en que se ubica dicho elemento [5].

2.4.1. Elementos de un sistema SCADA

2.4.1.1. Hardware SCADA

En la parte de hardware, SCADA se compone de dos tipos de estaciones, las unidades terminales remotas (RTU, por sus siglas en inglés) y unidades terminales maestras (MTU, por sus siglas en inglés), estos dos tipos de estación interactúan por medio de un sistema de comunicación [31]. En la Figura 2.7 se muestran elementos típicos en una arquitectura de un sistema SCADA.

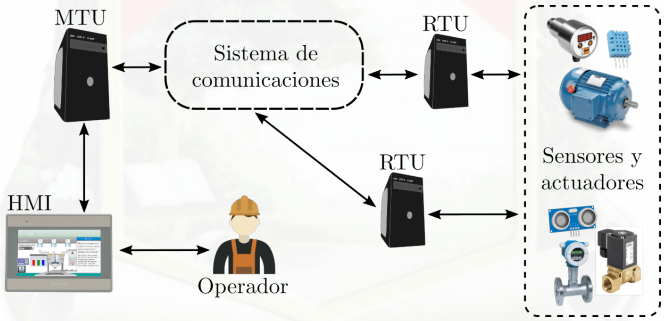


Figura 2.7: Ejemplo de una arquitectura de hardware en un sistema SCADA [5]

Las MTUs son la interfaz con el operador, además poseen la capacidad de comunicación con las RTUs. La función de las MTUs es mostrar la información recolectada por las RTUs y permite al operador ejecutar tareas de control de manera remota [5].

Las RTUs son la interfaz con los dispositivos de campo. La localización de las RTU suele ser cercana a los dispositivos de campo como sensores y actuadores, lo cual generalmente implica que se encuentren ubicadas lejos de las MTUs. Las RTU se encargan de la recolección de datos provenientes de los dispositivos de campo, además poseen un medio de comunicación con las MTU para el envío de la información que recolectan y para recibir/ejecutar las órdenes emitidas por un MTU [5]. En la Figura 2.8 se muestran ejemplos de interacciones entre la RTU con la MTU y los dispositivos de campo. Se observa que los dispositivos de campo manejan señales

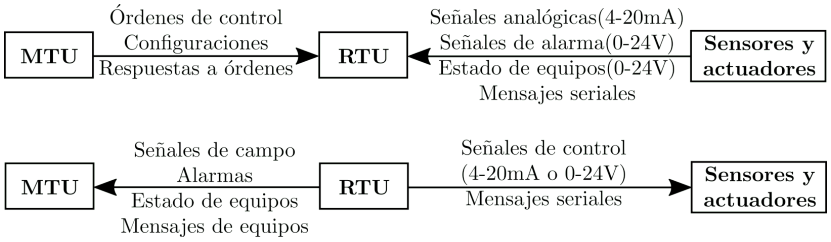


Figura 2.8: Ejemplos de mensajes intercambiados entre los componentes de un sistema SCADA [5]

analógicas que indican señales medidas, alarmas o señales de control, mientras que la MTU intercambia esta información con las RTU por medio de mensajes propios del protocolo de comunicación implementado.

El sistema de comunicación se encarga del intercambio de información entre las RTU y las MTU. La implementación de protocolos y métodos de detección de errores permite una comunicación más eficiente. El medio físico a utilizarse puede ser cobre, fibra óptica, radio, línea telefónica, microondas o por satélite [31].

2.4.1.2. Software SCADA

El sistema SCADA se presenta como un paquete de software que interactúa con el hardware [32]. El elemento principal en el software SCADA es el servidor, el cual se encarga de la adquisición y manejo de datos seleccionados en un sistema [32]. El software SCADA puede ser de tipo abierto o propietario. El software propietario es desarrollado por compañías para comunicarse con su propio hardware, lo cual crea una dependencia hacia los productos que oferta dicha compañía. Por otra parte, las soluciones de plataforma abierta ofrecen una alternativa para brindar interoperabilidad con dispositivos de distintos fabricantes [31].

2.4.1.3. Aplicación de SCADA en IoT

Dentro del desarrollo de la industria se distinguen cuatro grandes revoluciones. La primera revolución industrial surge a partir de la invención de la máquina de vapor que permitió aumentar la escala de producción. La segunda revolución industrial introduce la producción en serie gracias a las líneas de ensamblaje impulsadas por energía eléctrica. La tercera revolución industrial obedece a la automatización de procesos gracias al desarrollo de PLCs, en esta etapa surgen los sistemas SCADA como una solución para el monitoreo y control remoto de procesos de producción [33]. La cuarta revolución industrial, denominada Industria 4.0, propone la creación

de industrias inteligentes, de manera que las empresas implementen sistemas ciberfísicos que permitan: la conexión y control de sus equipos e instalaciones y el intercambio de información a través de redes globales [30].

En este contexto la aplicación de los conceptos del *Internet* de las cosas (IoT, por sus siglas en inglés) dentro de la industria surgen como la evolución natural de los sistemas SCADA. Sin embargo, las empresas no han aplicado ampliamente estos conceptos debido a que ya poseen sistemas SCADA desarrollados. Por lo tanto, la tendencia es implementar IoT a partir de los sistemas SCADA. A su vez, el desarrollo tecnológico provoca que los sistemas SCADA implementen nuevas tecnologías y protocolos para comunicación con sensores, así como la adopción de nueva tecnología *web* para permitir la integración con dispositivos inteligentes y la incorporación de nuevos métodos para el manejo de información. De esta manera los sistemas SCADA evolucionaron a plataformas para IoT y su desarrollo se encamina en esa dirección, manteniendo su rendimiento y estabilidad con la ventaja de permitir la incorporación de nueva tecnología y del IoT [33].

2.5. Comunicación industrial

2.5.1. Modelos de la pila de una red de comunicación

Gran parte de los protocolos de comunicación desarrollados no implementan todas las capas descritas en el modelo de interconexión abierta de sistemas (OSI, por sus siglas en inglés) ya sea porque son innecesarias para ciertos protocolos o se eligen combinar las características de una o más capas con el fin de simplificar la implementación del protocolo como se muestra en la Figura 2.9. Por ejemplo, el protocolo TCP/IP consiste solo de la capa física, de red, de transporte y de aplicación. TCP/IP a pesar de consistir de cuatro capas es un protocolo completamente funcional y se usa en Internet y como base para protocolos en tiempo real sobre *Ethernet*. Por otro lado, gran parte de los buses seriales de campo necesitan solo de tres capas para su implementación, la capa física, la capa de enlace de datos y la capa de aplicación [6].

2.5.2. Protocolos de comunicación industrial

A medida que la automatización de los procesos industriales ha evolucionado se han desarrollado una gran cantidad de protocolos de comunicación entre los mismos. Tradicionalmente, los protocolos basados en topologías de bus han sido los más aceptados dentro de las redes industriales. Las primeras generaciones de los mismos fueron desarrollados sobre estándares de comunicación serial como RS-232 y RS-485 [34]. Los protocolos industriales de bus serial más

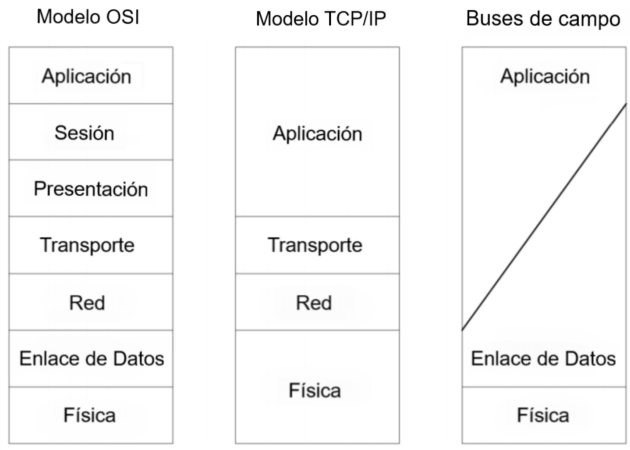


Figura 2.9: Modelos de la pila de una red de comunicación [6]

comunes son:

- Profibus [35]: Es uno de los protocolos más conocidos debido a que cuenta con el respaldo de *Siemens*. Todas las variantes de este protocolo implementan el acceso de bus mediante paso de testigo (token-pass, por su traducción al inglés) y con múltiples maestros con capacidad de sondear a otros dispositivos [6].
- *Modbus RTU* [36]: Es un protocolo que se basa en el paradigma maestro-esclavo. Es decir, solo el dispositivo maestro puede iniciar consultas, mientras los esclavos solamente responden a dichas consultas con la información pertinente. *Modbus* se convirtió en un protocolo de facto en los dispositivos de automatización industrial [37].
- Red de área de control (CAN [38], por sus siglas en inglés): Es un protocolo desarrollado por *BOSCH* para su uso en automóviles. En el área de automatización industrial no es adecuado debido a su falta de funcionalidad de alto nivel, como la provisión de la capa de aplicación. Sin embargo ha sido la base para protocolos de alto nivel como *CANopen* [6].

Gran parte de los protocolos de bus serial tuvieron que ser modificados para soportar *Ethernet* y nuevos protocolos también fueron desarrollados debido a la gran popularidad que *Ethernet* alcanzó dentro de las redes industriales. *Ethernet* ganó gran interés dentro de la industria ya que permite una interconexión más sencilla entre las redes de oficina y las redes industriales para compartir información de los procesos y control industrial [6]. Además el desarrollo de *Ethernet* con tecnología de conmutación y dúplex completo ganó aún mayor interés para aplicaciones industriales en tiempo real [34]. Los protocolos más comunes basados en *Ethernet* Industrial y con mayor potencial de cumplir los requerimientos de aplicaciones industriales en tiempo real son [39]:

- *EtherNet/IP* [40]: Es un protocolo que hace uso del protocolo industrial común (CIP, Por sus siglas en inglés). CIP define los objetos y sus relaciones en diferentes perfiles y cumple con tiempos de reacción de alrededor de 100 ms [39].
- *PROFInet* [41]: Definido principalmente por *Siemens* y respaldado por *Profibus International* [39]. *PROFInet* es la adaptación de los modelos y versiones de *PROFIBUS* a *Ethernet*. *PROFINET* utiliza llamadas a procedimientos remotos (RPC, por sus siglas en inglés) y el modelo de objetos de componentes distribuidos (DCOM, por sus siglas en inglés) para comunicaciones en el rango de 50 ms a 100 ms [6].
- *Powerlink* [42]: Se basa en el principio de utilizar un sistema de programación maestro-esclavo en la parte superior de un segmento *Ethernet* común compartido. El maestro garantiza el acceso en tiempo real a los datos cíclicos y permite el paso de tramas TCP/IP estándar solo en intervalos de tiempo específicos [39].
- *Modbus/TCP* [36]: Es una adaptación del *Modbus* clásico sobre una red TCP/IP. Este protocolo es probablemente la solución actualmente más utilizada sobre redes *Ethernet* para aplicaciones industriales y cumple con tiempos de reacción de alrededor de 100 ms sin problemas [39]. Por esta razón, en la siguiente subsección se presenta una descripción general del protocolo *Modbus*.

2.5.3. Protocolo Modbus

Modbus es un protocolo de comunicación implementado en la capa de aplicación, posicionado en el nivel 7 del modelo OSI. Este protocolo proporciona comunicación del tipo cliente/servidor entre dispositivos conectados a diferentes tipos de buses o redes. *Modbus* es un protocolo de solicitud/respuesta y ofrece servicios especificados por códigos de función. Actualmente *Modbus* se implementa utilizando redes TCP/IP sobre *Ethernet*, redes con transmisión serial asincrónica como RS-232 y RS-485 o en redes con paso de testigo en una red de alta velocidad conocida como *Modbus plus* [7]. Los modelos en capa de las distintas versiones de *Modbus* se resumen en la Figura 2.10.

2.5.3.1. Descripción del protocolo Modbus

La trama de *Modbus* (Figura 2.11) define una unidad de datos de protocolo (PDU, por sus siglas en inglés) muy simple. La PDU está formada por un código de función y un campo de datos. El mapeo del protocolo *Modbus* en buses o redes específicas puede introducir algunos campos adicionales en la unidad de datos de la aplicación (ADU, por sus siglas en inglés) [7].

La PDU *Modbus* está construida por el cliente que inicia una transacción *Modbus*. El campo de código de función contiene un código de función o de sub-función que indica al servidor qué

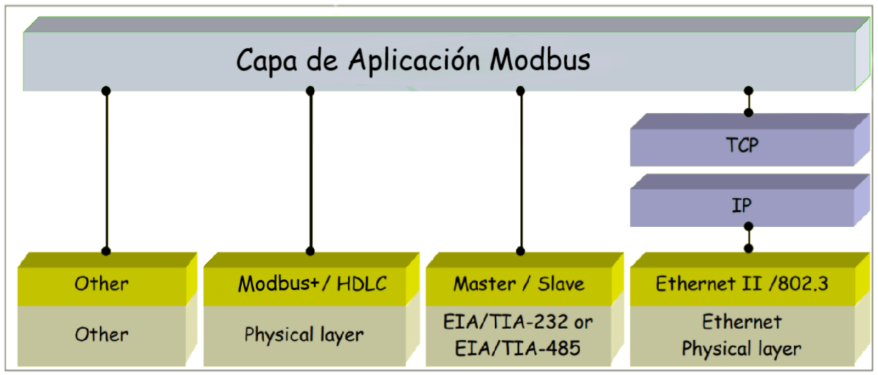


Figura 2.10: Implementación de *Modbus* en modelo de capas [7]

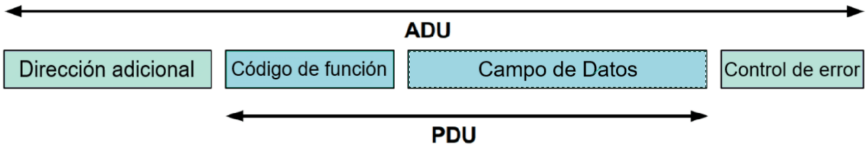


Figura 2.11: Trama de *Modbus* [7]

tipo de acción realizar. El campo del código de función de una PDU de *Modbus* está codificado en un byte. Los códigos válidos están en el rango de 1 a 255, el rango de 128 a 255 está reservado para respuestas de excepción. El código de función "0" no es válido. El campo de datos contiene información adicional que el servidor usa para tomar la acción definida por el código de la función. Esto puede incluir elementos como direcciones de datos discretos y de registro, la cantidad de elementos a manipular y el recuento de bytes de la carga útil en el campo. El campo de datos puede ser inexistente en ciertos tipos de solicitudes, en este caso el servidor no requiere ninguna información adicional. Si no se produce ningún error con la función solicitada, el servidor envía una PDU de respuesta con los datos solicitados en el campo de datos. El proceso de comunicación *Modbus* se muestra en la Figura 2.12 [7].

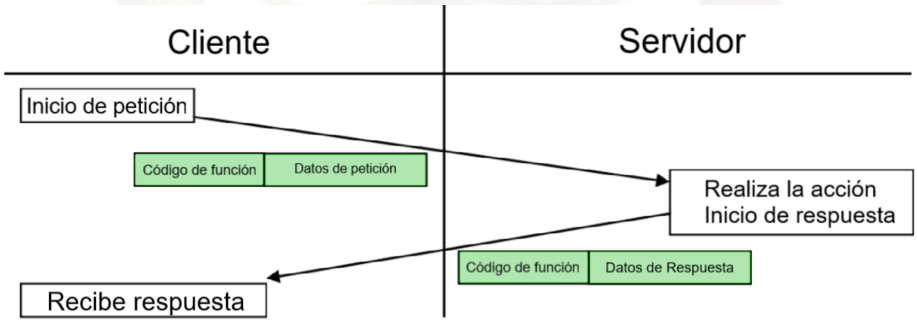


Figura 2.12: Transacción del protocolo *Modbus* sin errores [7]

Si se produce un error relacionado con la función *Modbus* solicitada, el campo de código de función contiene un código de función de error indicando el tipo de error ocurrido. Por su parte, el campo de datos contiene un código de excepción que la aplicación del servidor puede usar para determinar la siguiente acción a realizar (Figura 2.13) [7].

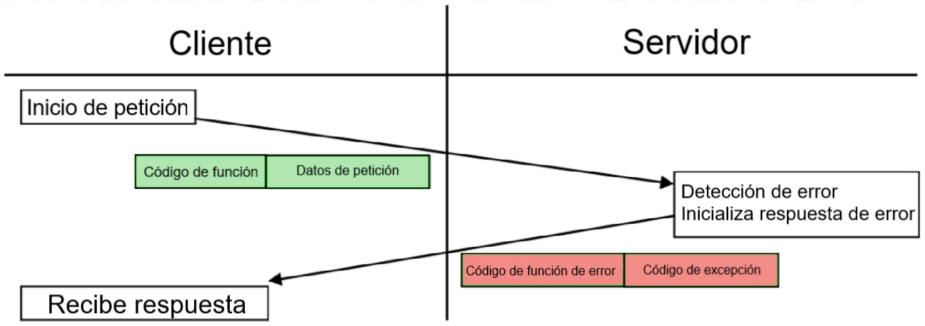


Figura 2.13: Transacción del protocolo *Modbus* cuando existe un error [7]

Modbus tiene tres clases de códigos de función, que son:

- Códigos de función públicos: Son códigos de función bien definidos, con garantía de ser únicos, validados por la comunidad de *modbus.org* [36] y documentados públicamente. Por ejemplo, códigos de función para funciones de lectura y escritura de datos.
- Códigos de función definidos por el usuario: Son códigos de función libres que el usuario puede seleccionar para implementar una función que no es compatible con la especificación de *Modbus*. Los códigos libres están dentro de los rangos de 65 a 72 y de 100 a 110. No está garantizado de que el código de función sea utilizado por una única función.
- Códigos de función reservados: Son códigos de función utilizados por algunas compañías para productos heredados y que no están disponibles para uso público.

2.5.3.2. Modelo de datos del protocolo Modbus

Modbus usa un modelo de datos basado en tablas de datos, Cada tabla tiene características diferentes. Los tipos de tablas de datos que maneja *Modbus* son descritos en la Tabla 2.1.

Para cada una de las listas principales, el protocolo permite la selección individual de 65536 elementos de datos, y las operaciones de lectura o escritura de esos elementos están diseñadas para abarcar múltiples elementos de datos consecutivos hasta un límite de tamaño dependiendo del tipo de código de función [7].

Tabla 2.1: Modelo de datos del protocolo *Modbus*

Tipos de tabla	Tamaño	Acceso
Entradas discretas	1 bit	Solo lectura
Coils	1 bit	Lectura y escritura
Registros de entrada	16 bits	Solo lectura
Registros de retención	16 bits	Lectura y escritura

2.6. Algoritmos de control

2.6.1. Esquema de control proporcional-integral-derivativo

En la actualidad gran cantidad de los procesos de control industrial utilizan esquemas basados en el controlador proporcional-integral-derivativo (PID, por sus siglas en inglés) debido a su simplicidad de implementación y robustez ante una amplia gama de condiciones. La función de transferencia del controlador PID está definida por la Ecuación 2.1 [8].

$$G_s(s) = \frac{K_p + K_i}{s + K_d s} \tag{2.1}$$

Donde:

- K_p es la contante proporcional
- K_i es la constante integral
- K_d es la constante derivativa

El término K_p realiza la acción de control proporcional. Este tipo de control trata de reducir la señal de error en estado estacionario. Por otro lado el término K_i/s realiza la acción de control integral, el cual elimina la desviación estacionaria de la señal de error remanente del control proporcional. Mientras el término $K_d s$ realiza la acción de control derivativo que mejora la velocidad de corrección del error utilizando la tasa de cambio de la señal de error [43].

La expresión del control PID en tiempo discreto esta definido por la Ecuación 2.2.

$$u_k = K_p e_k + K_i T_s \sum_{i=1}^{n} e_i + \frac{K_d}{T_s} \Delta e_k \tag{2.2}$$

Donde:

- u_k es la señal de control
- e_k es la señal de error
- $\Delta e_k = e_k - e_{k-1}$

2.6.2. Esquema de control PID con programación de ganancia difusa

El esquema de control PID con programación de ganancia difusa (FGS-PID, por sus siglas en inglés) es un control PID auto-ajustable [8]. Este esquema de control explota las reglas difusas y el razonamiento aproximado para ajustar las ganancias proporcional, integral y derivativa del control PID. Las ganancias del control PID son calculadas mediante las Ecuaciones 2.3-2.5:

$$K_p = (K_{p,max} - k_{p,min})K'_p + K_{p,min} \tag{2.3}$$

$$K_d = (K_{d,max} - K_{d,min})K'_d + K_{d,min} \tag{2.4}$$

$$K_i = \frac{{K_p}^2}{\alpha K_d} \tag{2.5}$$

donde K'_p, K'_d y α son obtenidos mediante reglas difusas utilizando las señales de error $e(k)$ y la primera derivada de la señal de error $e(\dot{k})$. Las reglas difusas tienen la forma:

$$\begin{aligned} &If\ e(k)\ is\ A_i\ and\ \dot{e}(k)\ is\ B_i,\ then\ K'_p\ is\ C_i\ and\ K'_d\ is\ D_i\ and\ \alpha = \alpha_i \\ &Para\ i = (1, 2, 3, ..., M) \end{aligned} \tag{2.6}$$

donde A_i, B_i, C_i y D_i son conjuntos difusos de soporte para las funciones de membresía.

Las formas de las funciones de membresía de los conjuntos difusos A_i y B_i para $e(k)$ y $\dot{e}(k)$ se muestran en la Figura 2.14. Donde la letra N representa valores negativos, P representa valores positivos, B representa valores grandes, M representa valores medios, S representa valores pequeños y ZO representa valores cercanos a cero.

Por otro lado las funciones de membresía para C_i y D_i para los valores de K'_p y K'_d tienen la forma que se muestra en la Figura 2.15. Para realizar la defusificación se usa el método del centroide [8].

El parámetro α también tiene una función de membresía la cual se muestra en la Figura 2.16.

En base a las funciones de membresía se determinan las reglas para la obtención de K'_p, K'_d y α. Estas reglas son resumidas en las Tablas 2.2, 2.3 y 2.4 respectivamente.

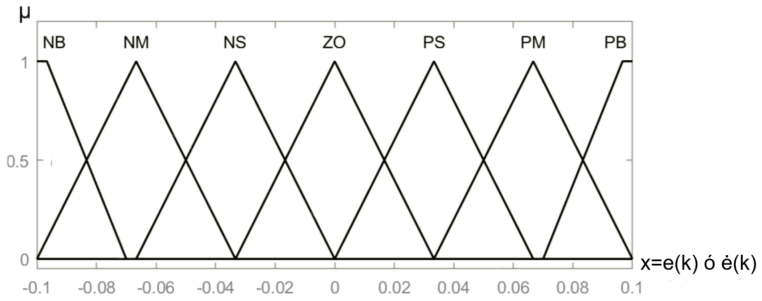


Figura 2.14: Funciones de membresía para $e(k)$ y $\dot{e}(k)$ [8]

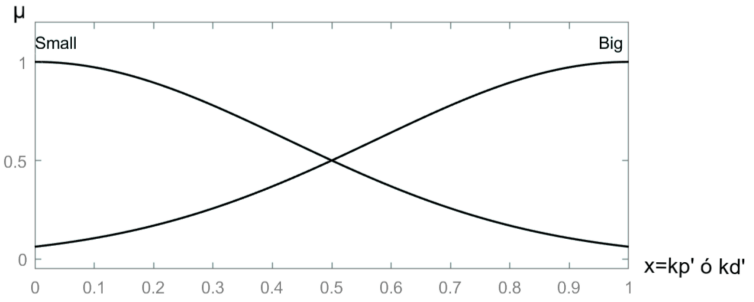


Figura 2.15: Funciones de membresía para k'_p y k'_d [8]

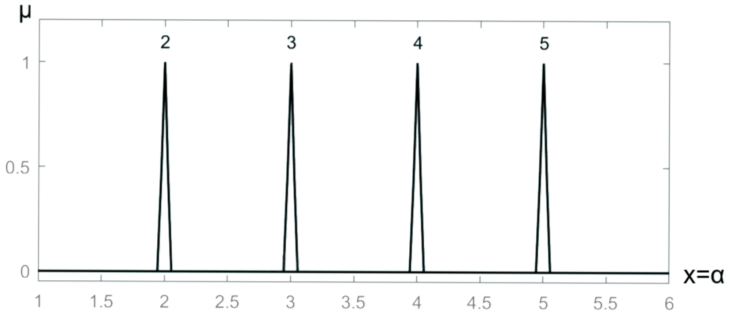


Figura 2.16: Funciones de membresía para α [8]

Tabla 2.2: Reglas de ajuste para Kp' [8]

		$\dot{e}(k)$						
		NB	NM	NS	ZO	PS	PM	PB
	NB	B	B	B	B	B	B	B
	NM	S	B	B	B	B	B	S
	NS	S	S	B	B	B	S	S
$e(k)$	ZO	S	S	S	B	S	S	S
	PS	S	S	B	B	B	S	S
	PM	S	B	B	B	B	B	S
	PB	B	B	B	B	B	B	B

Tabla 2.3: Reglas de ajuste para Kd' [8]

		$\dot{e}(k)$						
		NB	NM	NS	ZO	PS	PM	PB
	NB	S	S	S	S	S	S	S
	NM	B	B	S	S	S	B	B
	NS	B	S	B	S	B	B	B
$e(k)$	ZO	B	B	B	B	B	B	B
	PS	B	B	B	S	B	B	B
	PM	B	B	S	S	S	B	B
	PB	S	S	S	S	S	S	S

Tabla 2.4: Reglas de ajuste para α [8]

		$\dot{e}(k)$						
		NB	NM	NS	ZO	PS	PM	PB
	NB	2	2	2	2	2	2	2
	NM	3	3	2	2	2	3	3
	NS	4	3	3	2	3	3	4
$e(k)$	ZO	5	4	3	3	3	4	5
	PS	4	3	3	2	3	3	4
	PM	3	3	2	2	2	3	3
	PB	2	2	2	2	2	2	2

2.7. Observador de estados

2.7.1. Filtro de Kalman

El filtro de Kalman (KF, por sus siglas en inglés) es un observador de estados muy poderoso que permite estimaciones de estados pasados, presentes y futuros incluso cuando es desconocido el modelo preciso del sistema [9]. Este filtro es aplicable en sistemas lineales definidos por las Ecuaciones 2.7 y 2.8.

$$\bar{x}_k = A\hat{x}_{k-1} + Bu_{k-1} + w_{k-1} \tag{2.7}$$

$$z_k = Hx_k + v_k \tag{2.8}$$

donde w_k y v_k son variables aleatorias que representan el ruido presente en el sistema.

El KF usa un algoritmo computacional recursivo definido por un conjunto de ecuaciones matemáticas. Las ecuaciones que definen al KF se dividen en dos grupos: las ecuaciones de actualización de tiempo y ecuaciones de actualización de medición [9]. Estas ecuaciones son:

$$\bar{x}_k = A\hat{x}_{k-1} + Bu_{k-1} \tag{2.9}$$

$$\bar{P}_k = AP_{k-1}A' + Q \tag{2.10}$$

$$K_k = \bar{P}_k H'(H\bar{P}_k H' + R)^{-1} \tag{2.11}$$

$$\hat{x}_k = \bar{x}_k + K_k(z_k - H\bar{x}_k) \tag{2.12}$$

$$P_k = (I - K_k H)\bar{P}_k \tag{2.13}$$

Donde:

- $\bar{x}$, $\hat{x}$: es la estimación del estado a priori y posteriori respectivamente.
- A: es la matriz que relaciona el estado x_k con el estado x_{k-1} en ausencia de ruido de proceso
- B: es la matriz que relaciona el estado x con la señal de control u
- H: es la matriz de estados que relaciona el estado con la salida del proceso o medición
- Q: es la covarianza del ruido del proceso
- R: es la covarianza del ruido de las mediciones
- $\bar{P}$,P: son las covarianzas del error a priori y posteriori respectivamente.

- K: es la ganancia del filtro del Kalman

Las ecuaciones de actualización de tiempo o predicción: 2.9 y 2.10 tienen la función de proyectar las estimaciones de covarianza del estado actual y del error para obtener una estimación a priori del siguiente estado. Por otro lado las ecuaciones de actualización de medición o corrección: 2.11, 2.12 y 2.13 son las que realizan la retroalimentación para obtener una estimación mejorada a posteriori a partir de la estimación a priori [9]. Estos procesos pueden visualizarse de manera ilustrativa en la Figura 2.17.

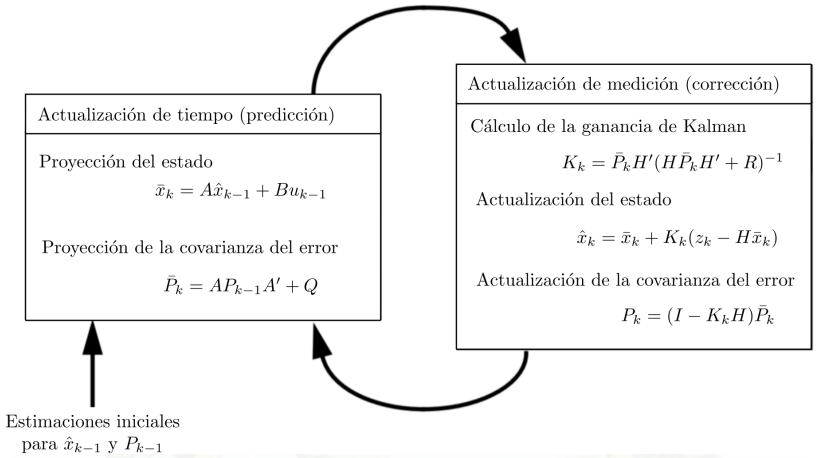


Figura 2.17: Algoritmo del filtro de Kalman [9]

2.7.2. Filtro de Kalman extendido

El filtro de Kalman extendido (EKF, por sus siglas en inglés) es una variación del filtro de Kalman para sistemas no lineales definidos por las ecuaciones 2.14 y 2.15.

$$\bar{x}_k = f(x_{k-1}, uk - 1, w_{k-1}) \tag{2.14}$$

$$z_k = h(x_k, v_k) \tag{2.15}$$

El EKF linealiza el sistema sobre la media actual aplicando el Jacobiano para obtener las matrices de estado A_k, H_k, W_k y V_k. En este caso W_k y V_k son los Jacobianos calculados con respecto a las variables aleatorias w_k y v_k respectivamente [9]. Con estas consideraciones las ecuaciones que definen el EKF son:

$$\bar{x}_k = f(\hat{x}_k - 1, u_{k-1}, 0) \tag{2.16}$$

$$\bar{P}_k = A_k P_{k-1} A'_k + W_k Q_k W'_k \tag{2.17}$$

$$K_k = \bar{P}_k H'_k (H_k \bar{P}_k H'_k + R_k)^{-1} \tag{2.18}$$

$$\hat{x}_k = \bar{x}_k + K_k (z_k - H_k \bar{x}_k) \tag{2.19}$$

$$P_k = (I - K_k H_k) \bar{P}_k \tag{2.20}$$

Capítulo 3

Estado del arte

Este capítulo presenta un resumen de propuestas relacionadas en el presente trabajo y enmarcada en el contexto de la Industria 4.0. En dichas propuestas se analizan protocolos, arquitecturas de sistemas de comunicaciones industriales, y el uso de software y hardware en sistemas de control industrial, entre otros.

En [44] se propone un sistema para monitorear el flujo de un líquido por medio de una red de área local (LAN, por sus siglas en inglés). Para ello se hace uso de una arquitectura formada por sensores, actuadores, un microcontrolador, un microcomputador y un servidor web. Como microcontrolador se utiliza un *Arduino*, el cual mide el flujo del líquido utilizando un sensor de efecto *Hall* y controla una electroválvula. El *Arduino* a su vez se encuentra conectado a un *Raspberry Pi*, el mismo que actúa como microcomputador y tiene configurado el servidor web, para controlar de manera remota la electroválvula y monitorear el flujo del líquido en el sistema a través de la red LAN.

En [45] se propone el diseño de sistemas SCADA por medio de herramientas de hardware y software libre. La estructura SCADA planteada por el autor se compone de 4 elementos: una unidad terminal maestra (MTU, por sus siglas en inglés), el protocolo de comunicación, una unidad terminal remota (RTU, por sus siglas en inglés) y los dispositivos de campo. Dentro de esta estructura básica, el MTU se encuentra descrito por un ordenador con un interfaz humano-máquina (HMI, por sus siglas en inglés), el RTU es una plataforma embebida basada en Arduino, por otra parte el protocolo de comunicación entre el MTU y RTU es *Modbus RTU*. Tomando como referencia la estructura básica descrita anteriormente, el autor implementa una red SCADA con dos RTUs para el monitoreo del nivel de líquido de dos tanques. El primer RTU utiliza un sensor de flujo y una válvula para el control del primer tanque. El segundo RTU utiliza un sensor de flujo y dos válvulas para el control del segundo tanque. Tanto la comunicación entre los RTU y el MTU, como la comunicación entre la interfaz *HMI* y la planta, se implementó por medio de *Modbus RTU* [36]. El diseño del interfaz HMI se realizó por medio de la aplicación *Advanced HMI* sobre *Visual Studio*. La función de la interfaz HMI es la visualización del nivel de los tanques y estado de las válvulas.

En [46] se presenta un banco de pruebas formado por 6 *Raspberry Pi* para ilustrar un método de resolución de problemas de protocolos de comunicación en sistemas industriales inteligentes dentro del paradigma de Industria 4.0 o Internet industrial de las cosas (IIoT, por sus siglas en inglés).

En este mismo contexto de Industria 4.0 y utilizando las mismas plataformas embebidas para su evaluación, en [47] se propone una arquitectura para un sistema multi-agente basado en OPC-UA para la integración de sistemas en emplazamientos de manufactura. Adicionalmente, como ejemplos de uso de estas plataformas de bajo costo en sistemas de robótica industrial se pueden mencionar las reportadas en [48] y [49].

En [10] se propone la arquitectura presentada en la Figura 3.1 basada en redes definidas por software (SDN, por sus siglas en inglés). Esta arquitectura se compone de tres capas:

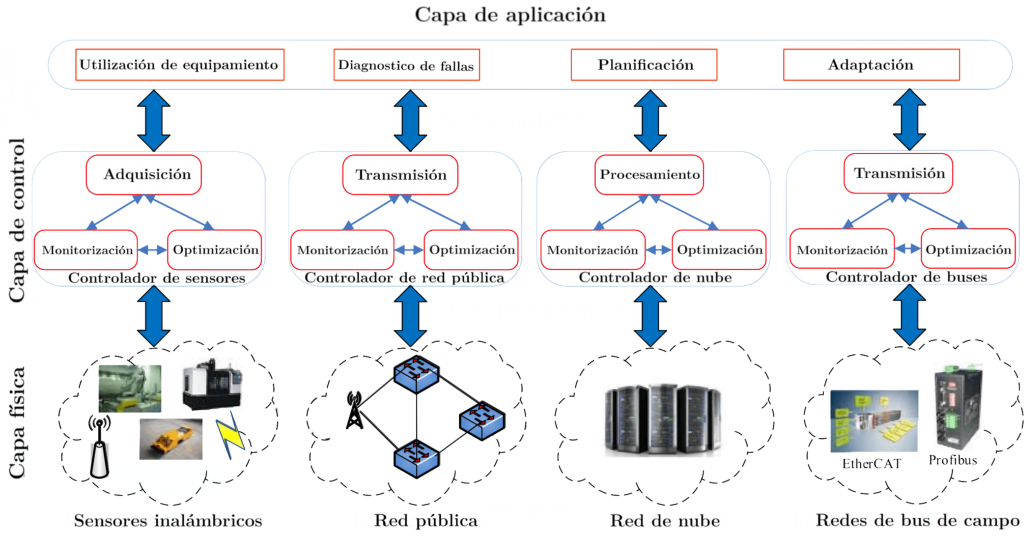


Figura 3.1: Arquitectura de IIoT definido por software [10]

- Capa de infraestructura física: En esta capa se incluye todo tipo de dispositivos, redes inalámbricas, redes de bus de campo, redes de *core*, hasta redes de nube y niebla.
- Capa de control: Esta capa realiza la interacción entre la capa de aplicación y la capa de infraestructura física mediante controladores de equipamiento y redes. Esta capa maneja todo el equipamiento y redes físicas adaptando todas sus funciones de acuerdo a los requerimientos de desempeño. Dentro del paradigma de la Industria 4.0 la capa de control realiza las funciones de recolección, transmisión y procesamiento de información.
- Capa de aplicación: Esta capa provee interfaces de programación de aplicaciones (API, por sus siglas en inglés) para la creación de varias aplicaciones con funciones de monitoreo, procesos de producción, entre otros.

Esta arquitectura presenta un mejor rendimiento tanto en eficiencia como en consumo energético, pero su implementación resulta un reto en escenarios con un número alto de nodos lo cual requerida de múltiples controladores distribuidos, retrasos en el *forwarding* y perdidas de paquetes en una arquitectura centralizada, además de que la escalabilidad no es garantizada debido a la complejidad de la arquitectura de software de los controladores.

En [50] se propone una arquitectura orientada al servicio para aplicaciones industriales basada en servidores OPC. Esta arquitectura se compone de tres niveles. El primer nivel se compone de servidores OPC-UA que recopilan los datos de sensores y actuadores, estos a su vez aseguran la comunicación en tiempo real entre los dispositivos. El segundo nivel se compone por capas de servicio que garantizan la flexibilidad y adaptabilidad del sistema. En el último nivel se implementa un solucionador de problemas de satisfacción de restricción (CSP,por sus siglas en

inglés) el cual realiza la optimización y programación de los planes de producción. El fin de esta arquitectura es programar, optimizar y automatizar los procesos de producción para la fabricación de peticiones realizadas por usuarios clientes.

En [11] proponen un sistema SCADA de código libre basado en un cliente OPC-UA compatible con un servidor OPC-UA que soporta protocolos de comunicación industrial como *Modbus*, CAN, RS-232, RS-485. El servidor OPC-UA tiene la funcionalidad de adquirir información de los dispositivos de control de la planta y a su vez intercambiar información entre los dispositivos de control y el cliente SCADA. La arquitectura de este sistema se muestra en la Figura 3.2.

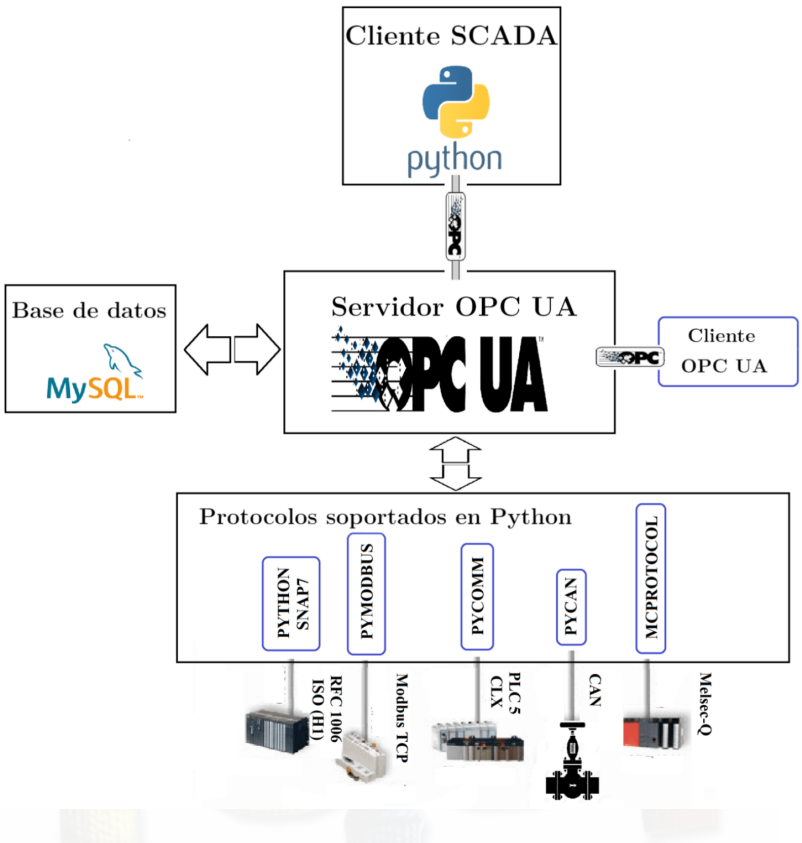


Figura 3.2: Arquitectura del sistema SCADA de código libre [11]

En [51] se presenta el desarrollo de una arquitectura de software libre para CIM avanzada (OSA-CIM, por sus siglas en inglés) utilizando *Java* y *Python*. La arquitectura propuesta se compone de tres elementos: un servidor OPC desarrollado en *Python*, un *software* SCADA desarrollado en *Java* y el cliente OPC-UA *UaExpert* [52] para demostrar interoperabilidad con clientes de otros desarrolladores. Las funcionalidades que presenta la arquitectura son comunicación, registro de datos y presentación de información al usuario. La comunicación entre clientes y servidor

se realiza mediante el estándar OPC-UA, por lo cual es multiplataforma y garantiza la interacción con *software* comercial que implemente dicho estándar de comunicación. El OSACIM fue probado en laboratorio mediante la emulación de la etapa de premolienda de una industria de fabricación de cemento.

En [53] se analiza la integración vertical en redes de comunicación industrial. Se plantean conceptos y criterios para el diseño de *gateways* que permitan la interconexión de redes de campo con redes de oficina basadas en IP. El *gateway* actúa como una interfaz hacia la red de campo. En el trabajo se analizan diferentes protocolos basados en IP, con el objetivo de aprovechar la interoperabilidad potencial con aplicaciones cliente existentes. Específicamente se analizan: SNMP, LDAP, SQL, tecnologías *web*, OPC XML-DA, OPC-UA, un protocolo propietario. El estándar OPC-UA se presenta como el más adecuado, ya que permite una adecuada representación del bus de campo, agregando acceso a datos históricos, alarmas y eventos. Adicionalmente brinda soporte para diferentes protocolos de comunicación, formatos de mensajes y servicios opcionales que lo hace adecuado para varios tipos de dispositivos, incluidos los sistemas embebidos.

En [54] se discute la implementación de SDNs en entornos CPS. En el trabajo se presentan las ventajas del uso de SDNs para aumentar la escalabilidad de redes, permitiendo el control de la red por parte de múltiples agentes externos coordinados. Adicionalmente se plantean los beneficios de la aplicación de conceptos como balanceo de carga, reservación de ancho de banda y virtualización de redes.

Capítulo 4

Desarrollo e implementación de los sistemas de instrumentación, comunicación y control

En este capítulo se trata de manera técnica la metodología empleada para el desarrollo e implementación de los sistemas de instrumentación, comunicación y control del sistema multi-tanque. Se presentan las características de los dispositivos utilizados, la descripción del diagrama de instrumentación de planta y la implementación de los sistemas de control y comunicación de la planta.

4.1. Descripción general del prototipo

Los sistemas desarrollados a lo largo de este trabajo tienen el objetivo de controlar los niveles de agua de un sistema multi-tanque, específicamente se controla el nivel de dos tanques de reserva. Para el control de los niveles se evalúan dos esquemas de control conformados por controladores PID clásicos y un FGS-PID. En adición, se propone un sistema de detección de fallas en el sistema multi-tanque mediante un EKF. El comportamiento del sistema de detección de fallas es evaluado en conjunto con los esquemas de control propuestos. Las fallas que se evalúan en este trabajo son presencias de fugas en cada tanque. Estas fugas son simuladas mediante un orificio ubicado en el fondo de cada tanque. Para esto se utilizan PLCs de bajo costo como son el *PiXtend* y *M-Duino*. También se implementa funciones de comunicación con una interfaz hombre-maquina programada en una pantalla *Siemens* para que los usuarios finales puedan supervisar y operar la planta. Adicionalmente, se implementa un servidor OPC-UA con el fin de adquirir datos y asegurar la interconexión entre la planta y aplicaciones clientes OPC-UA.

El propósito de este trabajo dentro del contexto del IIoT, cuya arquitectura es descrita en la Sección 2.1.2, es implementar la red de proximidad perteneciente al dominio de control y nivel de borde. El fin es que en trabajos futuros se puedan evaluar funcionalidades de IIoT sobre una planta compleja.

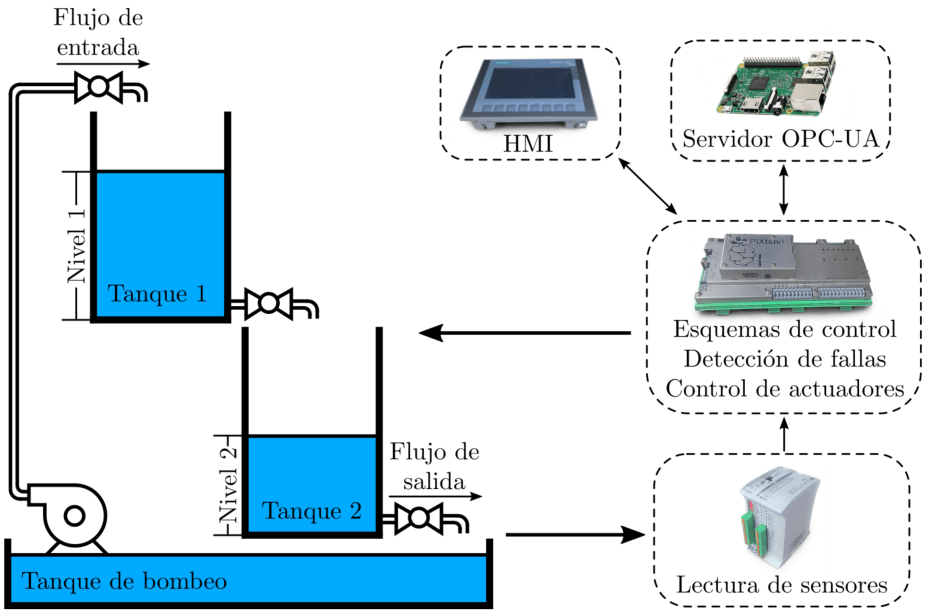


Figura 4.1: Esquema simplificado del sistema planteado en este trabajo

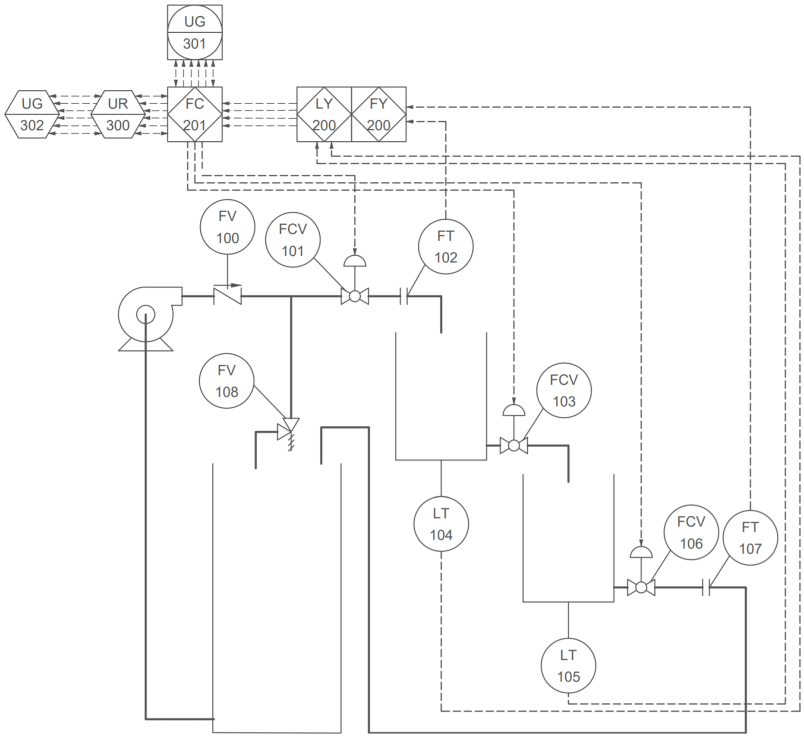


Figura 4.2: Diagrama P&ID de la planta del sistema multi-tanque

4.2. Diagrama de instrumentación y procesos del sistema multi-tanque

En la Figura 4.2 se presenta el diagrama de proceso e instrumentación (P&ID, por sus siglas en inglés) del sistema multi-tanque. El diagrama P&ID está realizado según la norma ISA-S5.1-84 [55].

En el diagrama P&ID se observa que la planta consiste de tres tanques, dos tanques de reserva y un tanque de bombeo. La bomba es protegida de golpes de ariete por una válvula mecánica *check* (FV 100). También se protege la bomba y las tuberías de incrementos de presión provocados por el cierre de la válvula FCV 101. Esto se realiza mediante una válvula de alivio de presión (FV 103) que regula la presión del sistema desfogando el agua de regreso al tanque de bombeo. El control de los niveles de los tanques de reserva se lo realiza mediante tres válvulas controlables (FCV 101, FCV 103 y FCV 106). Estas permiten la variación del flujo de entrada y salida de los dos tanques de reserva. El nivel de cada tanque de reserva es medido con un sensor de nivel (LT 104 y LT 105). Los flujos de entrada y salida del sistema multi-tanque son monitorizados mediante sensores de flujos (FT 102 y FT 107). Las señales eléctricas de los sensores de nivel y

flujo son preprocesados por un nodo auxiliar para obtener los valores numéricos de nivel y flujo (LY 200 y FY 200). Estos datos son enviados al nodo que realiza las funciones de controlador y observador (FC 201). Este a su vez envía datos pertinentes a un servidor de registro de datos (UR 300). Finalmente para supervisar la planta, se utiliza un interfaz hombre-maquina (UG 301), que se comunica directamente con el nodo controlador, y un SCADA (UG 302) que intercambia datos con el servidor de registro.

4.3. Componentes principales de los sistemas de instrumentación, comunicaciones y control de la planta

A continuación se describen brevemente los principales componentes utilizados para la implementación de los sistemas de instrumentación, comunicación y control.

4.3.1. Bomba

Para alimentar el sistema multi-tantque se utiliza la bomba Favson F3012 (Figura 4.3). Esta bomba trabaja nominalmente con una alimentación de 12 V para proporcionar un caudal de 4 L/min a una presión de 100 PSI. La entrada y salida de la bomba se adapta a tuberías de 3/8".

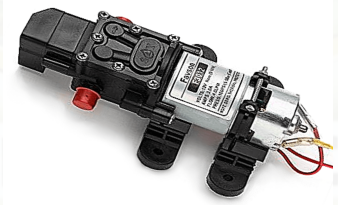


Figura 4.3: Bomba hidráulica Favson F3012

4.3.2. Electroválvulas

El control del sistema multi-tanque se realiza mediante la restricción de los flujos de entrada y salida de cada tanque. Para esto se utiliza las válvulas de control mostradas en la Figura 4.4. La válvula proporcional Winner WVA4-3 de tipo bola de 1/2" es utilizada para controlar el flujo de entrada de la planta. Esta válvula se alimenta con un voltaje nominal de 24 V y puede ser controlada mediante una señal de voltaje de 0 a 10 V o a través de una señal de corriente de 0 a 20 mA. La válvula BACOENG 2W-15 de tipo solenoide de 1/2" requiere ser alimentado con un

Figura 4.4: Válvulas de control

Figura 4.5: Módulos ultrasónicos HC-SR04 para medición de distancia

voltaje nominal de 12 V. Esta válvula es utilizada para controlar el flujo entre los dos tanques. La válvula U.S. SOLID USS-MSV00002 de tipo bola de 3/4" requiere una alimentación nominal de 24 V y su apertura es proporcional al tiempo de alimentación de la válvula. Esta válvula es utilizada para controlar el flujo de salida del sistema.

4.3.3. Sensores de nivel

Para monitorizar el nivel de los tanques se utilizan dos módulos ultrasónicos HC-SR04 que se muestran en la Figura 4.5. Estos módulos trabajan con una alimentación nominal de 5 V y permiten realizar mediciones de 2 cm a 4 m con un margen de error de ±3 mm. El módulo HC-SR04 mide el tiempo en que la onda ultrasónica retorna al reflejarse sobre un objeto. El cálculo de la distancia que recorre la onda ultrasónica hasta ser reflejada por un objeto está dado por la Ecuación 4.1.

$$distancia = \frac{t_{\text{medido}} * v_{\text{sonido}}}{2} \tag{4.1}$$

Donde t_{medido} es el valor medido por el sensor y v_{sonido}=340m/s.

Figura 4.6: Sensores de flujo de agua DIGITEN FL-408

4.3.4. Sensores de flujo

Los flujos de entrada y salida del sistema multi-tanque se monitorizan mediante los sensores de flujo DIGITEN FL-408 que se muestran en la Figura 4.6. Estos sensores trabajan con un voltaje de alimentación entre 5 V a 18 V y permiten medir flujos que se encuentran dentro del rango de 1 a 30 L/min y con una presión menor a 253.816 PSI. Las mediciones tienen un margen de error de $\pm 2\,\%$. El sensor de flujo proporciona una señal de pulsos cuya frecuencia es proporcional al flujo. El flujo en L/min puede ser calculado a partir de la Ecuación 4.2, la cual proviene de los datos proporcionados por el fabricante.

$$flujo = \frac{frecuencia\ de\ pulsos * 60}{7.5} \tag{4.2}$$

4.3.5. PLC *PiXtend V1.3*

El *PiXtend V1.3* (Figura 4.7) es un PLC basado en la plataforma *Raspberry Pi* como procesador auxiliar embebido. Consta de una gran cantidad de entradas y salidas digitales y analógicas que permiten conectar una amplia variedad de sensores y actuadores industriales. La ventaja adicional del *PiXtend* radica en su compatibilidad con plataformas de desarrollo/programación de software libre como Python. En el Apéndice C se encuentra la hoja de especificaciones técnicas del *PiXtend V1.3*. En la hoja de especificaciones del *PiXtend* se recomienda que el *PiXtend* trabaje con ciclos de trabajo de 100 ms y como mínimo soporta ciclos de trabajo de 25 ms. Esta restricción de tiempo se debe a la comunicación que se realiza entre el *Raspberry Pi* y el microcontrolador *Atmega32A* que controla los diferentes puertos de entrada/salida del *PiXtend*. Debido a estas restricciones de tiempo resulta imposible obtener las mediciones de los sensores HC-SR04 y DIGITEN FL-408, debido a que los pulsos de las señales de los sensores están en el rango de los ms y μs. Con el fin de solventar este problema, se utiliza el PLC *M-Duino* para que realice el pre-procesamiento de las señales de los sensores de nivel y flujo para obtener los valores de medición.

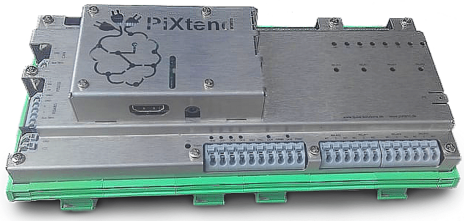


Figura 4.7: PLC *PiXtend V1.3*

Figura 4.8: PLC *M-Duino Ethernet 21 I/Os*

Estas restricciones permiten en este trabajo ilustrar la integración transparente de distintas plataformas embebidas de bajo costo, aprovechando la velocidad de manejos de entrada/salida que proporciona el *M-Duino* y la capacidad de procesamiento que presenta el *Pixtend*.

El *PiXtend* es encargado del control y la detección de fallos en el sistema multi-tanque.

4.3.6. PLC *M-Duino Ethernet 21 I/Os*

El *M-Duino Ethernet 21 I/Os* (Figura 4.8) es un PLC desarrollado sobre la plataforma de *Arduino MEGA*, posee 13 entradas y 8 salidas distribuidas entre digitales y analógicas y también es programable desde la IDE de *Arduino*. En el Apéndice D se encuentra la hoja de especificaciones técnicas del *M-Duino Ethernet 21 I/Os*. La función del *M-Duino* es realizar el pre-procesamiento de las señales de los sensores para obtener los valores de flujo y nivel de los tanques para enviarlos al *PiXtend* por medio de *Modbus*.

Figura 4.9: Pantalla KTP-700 Basic Color

4.3.7. Pantalla KTP-700 Basic Color

La pantalla *KTP-700 Basic Color* (Figura 4.9) es una pantalla industrial y táctil de 6" desarrollada por *Siemens*. Esta pantalla soporta los protocolos de comunicación industrial *Profinet* [41] y *Modbus TCP/IP* [36]. La pantalla *KTP-700 Basic Color* es programable desde *TIA PORTAL v14* de *Siemens*. La función de esta pantalla dentro de la planta es proveer un interfaz HMI para permitir la manipulación y visualización de la operación de la planta.

4.4. Diseño de la planta del sistema multi-tanque

En la Figura 4.11 se presenta el diseño 3D de la planta. Este se ha realizado en base al diagrama P&ID de la Figura 4.2. Las dimensiones de los tanques y del armazón de la planta se definieron en base a simulaciones, facilidad de construcción y a las dimensiones de los sensores y válvulas que se utilizan para el control y monitoreo de la planta. En el Apéndice A se presentan las láminas técnicas del armazón y de los tres tanques de la planta con sus respectivas dimensiones. En las láminas 2 y 3 del Apéndice A se muestra el diseño de los tanques de reserva. En estas láminas se especifica que los tanques tienen un pequeño agujero de desfogue en el fondo de cada tanque. Estos agujeros sirven para simular fallas en la planta debido a la presencia de fugas.

4.5. Diseño del tablero de instrumentación

El tablero de instrumentación se diseñó para facilitar el soporte de los dispositivos de control, comunicación, supervisión y fuentes de alimentación. El tablero dispone en su parte frontal de terminales de conexión para los sensores y actuadores de la planta, tres LEDs indicadores y la pantalla HMI. En la parte posterior se colocan todos los dispositivos de instrumentación y las canaletas de cableado. En las láminas 1 y 2 del Apéndice B se presenta el montaje del tablero de instrumentación y las dimensiones del tablero diseñado respectivamente. En la Figura 4.11

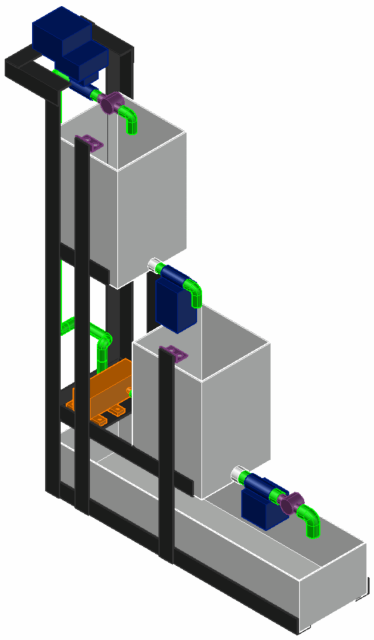


Figura 4.10: Diseño 3D de la planta del sistema multi-tanque

Tabla 4.1: Esquema de conexiones de los LEDs indicadores

Indicador	**Punto de conexión**
LED verde	GPIO1 del *PiXtend*
LED amarillo	GPIO0 del *PiXtend*
LED rojo	GPIO2 del *PiXtend*

se presenta el diseño 3D del tablero de instrumentación construido.

En la Figura 4.12 se muestra la forma en que se distribuyen los terminales de conexión y LEDs indicadores en el tablero de instrumentación. Se puede observar que los terminales de conexión de cada elemento de la planta se encuentran agrupados para facilitar al operador la instalación de los sensores y actuadores.

Los indicadores LED son conectados a los terminales GPIO del *PiXtend* conforme al detalle de la Tabla 4.1. El LED verde se utiliza para indicar que el sistema se encuentra funcionando en modo automático, por lo cual la bomba y válvulas son comandadas por medio del sistema de control planteado. El LED amarillo indica una fuga en el Tanque 1, mientras que el LED rojo indica una fuga en el Tanque 2. Cuando el sistema se encuentra en modo manual, el LED verde se apaga mientras que los LEDs amarillo y rojo se encuentran encendidos, en este modo el operador puede comandar la bomba y válvulas por medio del HMI.

En la Tabla 4.2 se detallan las conexiones internas del tablero con los PLCs.

El pin DO5 del *PiXtend* se configuró en modo PWM. Para la conexión de la válvula de bola se utilizó el circuito de adaptación presentado en la Figura 4.13. Esto es necesario debido a que las salidas digitales del *PiXtend* son conmutadas a GND, mientras que la válvula requiere que las señales para abrir o cerrar la válvula sean conmutadas a 24 V.

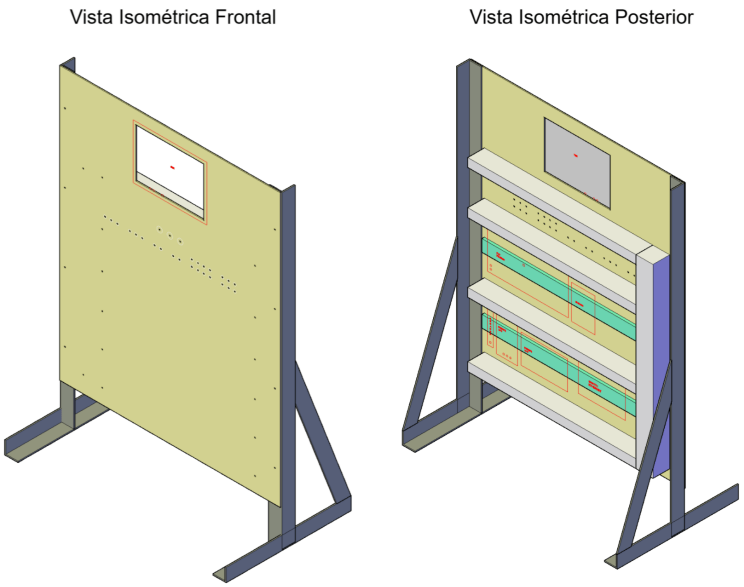


Figura 4.11: Diseño 3D del tablero de instrumentación

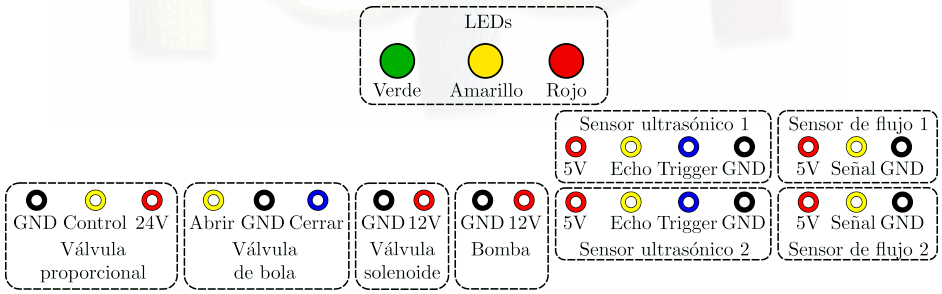


Figura 4.12: Distribución de terminales de conexión e indicadores de la parte frontal del tablero de instrumentación

Tabla 4.2: Esquema de conexión de los terminales ubicados en el tablero de instrumentación

Dispositivo	**Terminal de conexión**	**Punto de conexión**
Válvula proporcional (FCV 101)	GND	DO0 del *PiXtend*
	Control	AO0 del *PiXtend*
	24 V	Positivo de la fuente de alimentación de 24 V
Válvula de bola (FCV 106)	Abrir	DO1 del *PiXtend*
	GND	Negativo de la fuente de alimentación de 24 V
	Cerrar	DO2 del *PiXtend*
Válvula solenoide (FCV 103)	GND	DO5 del *PiXtend*
	12 V	Positivo de la fuente de alimentación de 12 V
Bomba	GND	Negativo de la fuente de alimentación de 12 V
	12 V	NC del Rele0 del *PiXtend*
Sensor ultrasónico 1 (LT 104)	5 V	5 Vdc del *M-Duino*
	Trigger	Q00 del *M-Duino*
	Echo	I00 del *M-Duino*
	GND	GND del *M-Duino*
Sensor ultrasónico 2 (LT 105)	5 V	5 Vdc del *M-Duino*
	Trigger	Q01 del *M-Duino*
	Echo	I01 del *M-Duino*
	GND	GND del *M-Duino*
Sensor de flujo 1 (FT 102)	5 V	5 Vdc del *M-Duino*
	Señal	PIN3 del *M-Duino*
	GND	GND del *M-Duino*
Sensor de flujo 2 (FT 107)	5 V	5 Vdc del *M-Duino*
	Señal	PIN2 del *M-Duino*
	GND	GND del *M-Duino*

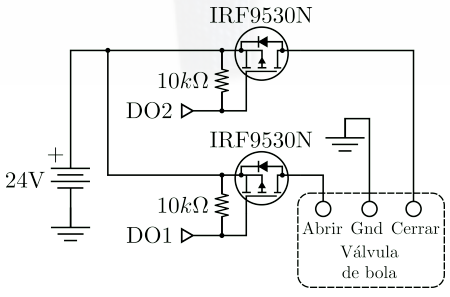


Figura 4.13: Circuito de adaptación para la válvula de tipo bola

4.6. Arquitectura del sistema de instrumentación y comunicación

En la Figura 4.14 se muestra la arquitectura de los sistemas de instrumentación y comunicación entre el nivel de campo, nivel de procesos y nivel de supervisión de acuerdo a la pirámide CIM. Desde el punto de vista del IIoT esta arquitectura pertenece al dominio de control, dentro del nivel de borde. A nivel de campo las mediciones de los sensores son interpretadas por el PLC *M-Duino*. Una vez que procesa las mediciones envía estos datos al PLC *PiXtend* mediante el protocolo *Modbus*. El protocolo *Modbus* es implementado en el *M-Duino* mediante la librería *mgsmodbus*. En tanto que en el *PiXtend* se implementa *Modbus* utilizando la librería *pymodbus*. El *PiXtend* se encarga de realizar el control de los niveles de los tanques de la planta mediante la regulación de las aperturas de las válvulas. También se encarga de realizar la detección de fallas en la planta. Las fallas son generadas mediante un orificio colocado en el fondo cada tanque a controlar para simular fugas. Por otra parte, el nivel de supervisión consta de la pantalla HMI *KTP-700* de *Siemens*, el servidor OPC-UA desarrollado en [56] y [11] por el equipo de trabajo del proyecto mencionado en 1.2, y un primer avance del sistema SCADA que se encuentra en desarrollo por el equipo de trabajo del proyecto. La comunicación entre el servidor OPC-UA y el *PiXtend* se realiza también a través del protocolo *Modbus* usando la librería *pymodbus*. El servidor OPC-UA tiene la funcionalidad de intercomunicar los datos de la planta hacia el SCADA. El OPC-UA también tiene la función de permitir la interconexión con clientes de nivel superior. Por otro lado, el *PiXtend* se comunica con el HMI *KTP-700* que soporta el protocolo *Modbus* para permitir la interactividad con el usuario que opera la planta.

4.6.1. Variables de proceso

En la arquitectura del sistema de comunicación se implementaron dos conjuntos de registros *Modbus*. Estos son generados por cada uno de los PLCs empleados en el sistema y contienen las variables de proceso. El conjunto de registros generados por el *M-Duino* corresponden a las mediciones de los sensores, este conjunto de variables se muestran en la Tabla 4.3.

Tabla 4.3: Variables de proceso generadas por el PLC *M-Duino*

Variable	Registro	Descripción
Tiempo de Tanque 1	40001	Tiempo en μs medido por el sensor ultrasónico del Tanque 1
Tiempo de Tanque 2	40002	Tiempo en μs medido por el sensor ultrasónico del Tanque 2
Flujo de entrada	40003	Flujo en lt/min medido a la entrada del sistema
Flujo de salida	40004	Flujo en lt/min medido a la salida del sistema

El conjunto de registros generados por el *PiXtend* corresponden a las variables empleadas en

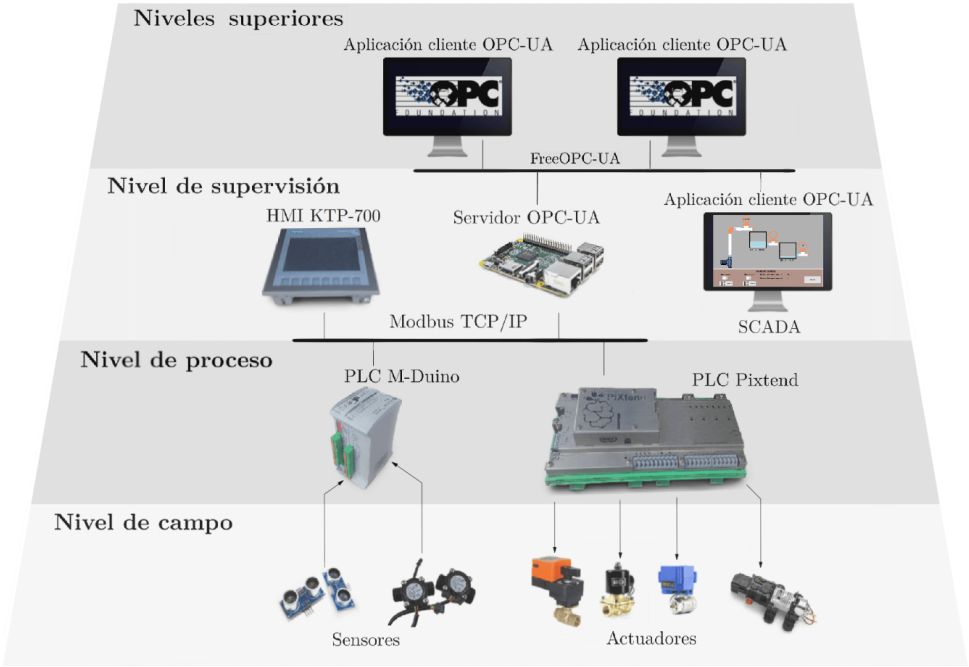


Figura 4.14: Arquitectura de la red de comunicación industrial

el proceso de control y detección de fallas, las mismas que son comunicadas tanto al HMI como al servidor OPC-UA. Este conjunto de variables se detallan en la Tabla 4.4.

Las variables de proceso descritas en las Tablas 4.3 y 4.4 utilizan los registros de retención de *Modbus*, los cuales manejan valores enteros positivos de 16 bits. Debido a esta limitación las señales de control y variables descritas tanto en m como lt/min son multiplicadas por 10000, mientras que las variables descritas en cm son multiplicadas por 100. Finalmente todas las variables de proceso son convertidas en enteros previo a su envío por *Modbus*.

4.6.2. Diseño del HMI

El HMI empleado para la visualización de las diferentes variables del proceso fue desarrollado en el software *TIA Portal V14* de *Siemens* para ser utilizado en el equipo *KTP-700 Basic Color*. El HMI consta de cinco pantallas seleccionables mediante los botones de F1 a F5 del equipo. En todas las pantallas se colocaron indicadores de alarma para mostrar fugas detectadas en cada uno de los tanques.

El botón F1 abre la pantalla principal del HMI, la cual presenta un diagrama del proceso. En la pantalla principal se tienen cuadros de texto para observar los valores del nivel de los tanques,

Tabla 4.4: Variables de proceso generadas por el PLC *PiXtend*

Variable	Registro	Descripción
Referencia del Tanque 1	40001	Nivel de referencia en cm para el Tanque 1
Referencia del Tanque 2	40002	Nivel de referencia en cm para el Tanque 2
Selector de tipo de controlador	40003	Determina si la Válvula 2 utiliza un control tipo PID o FGS-PID
Nivel del Tanque 1	40004	Nivel en m del Tanque 1
Nivel del Tanque 2	40005	Nivel en m del Tanque 2
Señal de control 1	40006	Porcentaje de apertura de la Válvula 1
Señal de control 2	40007	Porcentaje de apertura de la Válvula 2
Señal de control 3	40008	Porcentaje de apertura de la Válvula 3
Estimación de nivel del Tanque 1	40009	Nivel estimado en m para el Tanque 1
Estimación de nivel del Tanque 2	40010	Nivel estimado en m para el Tanque 2
Flujo de entrada	40011	Flujo en lt/min que ingresa al sistema
Flujo de salida	40012	Flujo en lt/min que sale del sistema
Estado de la bomba	40013	Determina si la bomba se encuentra encendida o apagada
Selector de modo de control	40014	Determina si el sistema funciona en modo manual o automático
Residuo de estimación del Tanque 1	40015	Diferencia en m entre el nivel medido y estimado en el Tanque 1
Residuo de estimación del Tanque 2	40016	Diferencia en m entre el nivel medido y estimado en el Tanque 2
Indicador de fuga en el Tanque 1	40017	Indica si existe una fuga en el Tanque 1
Indicador de fuga en el Tanque 2	40018	Indica si existe una fuga en el Tanque 2

señales de control para cada válvula, flujo de entrada y salida al sistema, además de indicadores visuales para los niveles de los tanques. También, se disponen de cuadros de texto para observar y modificar los valores de referencia. Además, se tiene selectores para escoger el tipo de control, entre Automático o Manual, y el tipo de controlador de la Válvula 2, entre PID o FGS-PID. Adicionalmente, se tiene un interruptor para encender o apagar la bomba. En la Figura 4.15 se muestra el resultado de la pantalla principal en un instante de operación del sistema.

El botón F2 por su parte, abre la pantalla de visualización de niveles, la cual presenta una gráfica temporal de los niveles medidos de los tanques y sus valores de referencia. El nivel del Tanque 1 se presenta en línea continua de color rojo, el nivel del Tanque 2 se presenta en línea continua de color azul, la referencia del Tanque 1 se presenta en línea discontinua de color rojo y la referencia del Tanque 2 se presenta en línea discontinua de color azul. En la Figura 4.16 se muestra el diseño desarrollado para la pantalla de visualización de niveles, la escala empleada se encuentra en cm.

Por otro lado, el botón F3 abre la pantalla de visualización de señales de control, la cual presenta una gráfica temporal de las señales de control que intervienen en el proceso. La señal de control

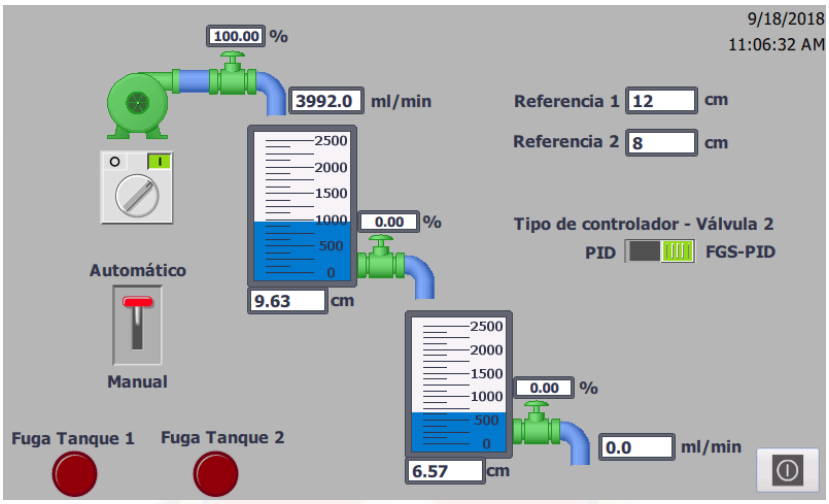


Figura 4.15: Pantalla principal del HMI

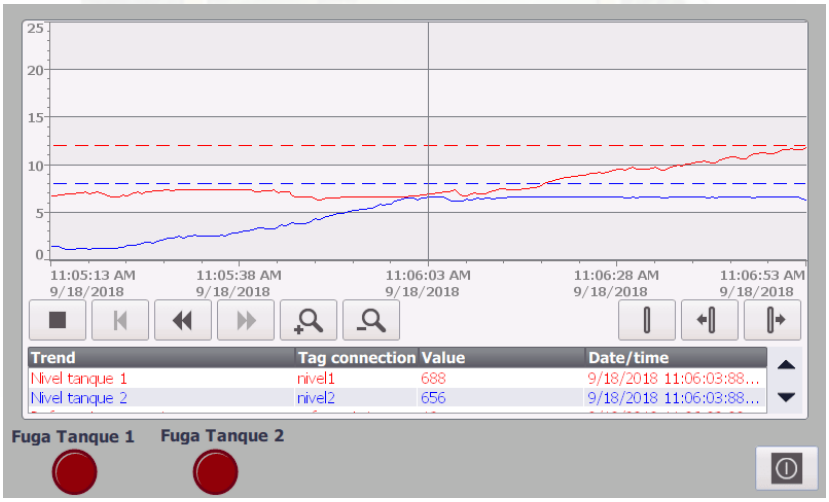


Figura 4.16: Pantalla de visualización de niveles

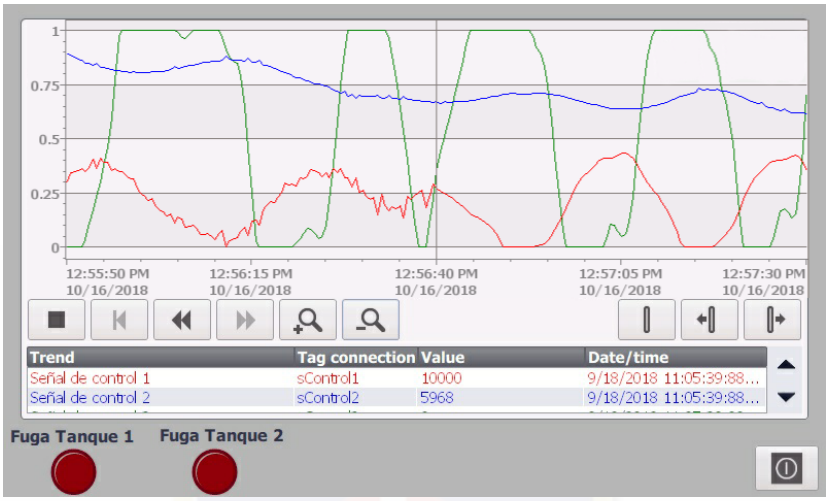


Figura 4.17: Pantalla de visualización de señales de control

de la Válvula 1 se presenta en línea continua de color rojo, la señal de control de la Válvula 2 se presenta en línea continua de color azul y la señal de control de la Válvula 3 se presenta en línea continua de color verde. En la Figura 4.17 se muestra el diseño desarrollado para la pantalla de visualización de señales de control, la escala empleada representa el porcentaje de apertura de las válvulas entre 0 y 1.

Finalmente, los botones F4 y F5 abren las pantallas de visualización del Tanque 1 y Tanque 2 respectivamente. Estas pantallas presentan una gráfica temporal del nivel medido, el nivel estimado y el residuo de la estimación para su respectivo tanque. El nivel medido se presenta en línea continua de color verde, el nivel estimado se presenta en línea discontinua de color azul y el residuo de estimación se presenta en línea continua de color rojo. En la Figura 4.18 se muestra el diseño desarrollado para la pantalla de visualización del Tanque 1, la escala ubicada a la izquierda se emplea para el residuo y se encuentra en metros, mientras que la escala a la derecha corresponde a los niveles y se encuentra en 1×10^4 m.

Por otra parte, el HMI se conecta por medio de *Modbus TCP* para leer los valores de las variables de proceso por medio de los registros de retención. Para permitir la comunicación por *Modbus*, el HMI se configuró con el *driver Modicon Modbus TCP/IP*, CPU de tipo *Concept, ProWORX: Compact, Quantum, Momentum*, servidor configurado con la dirección IP del *PiXtend* y puerto 502. Las variables de proceso se configuraron por medio de *tags* en el rango de direcciones desde 4x400002 hasta 4x400020. Esto corresponde a los registros de retención. El tiempo de adquisición es de 500 ms conforme al periodo de muestreo del sistema. En la Figura 4.19 se muestra el detalle de la asignación de *tags* para las variables de proceso empleadas.

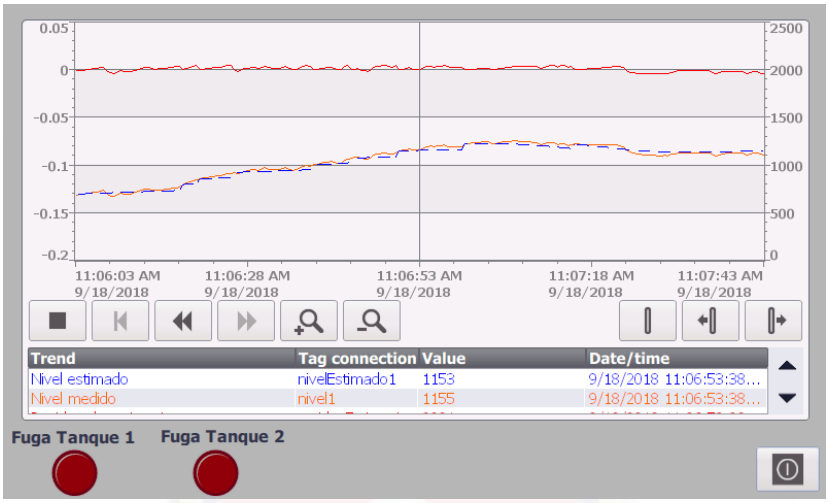


Figura 4.18: Pantalla de visualización del Tanque 1

HMI tags

Name	Data type	Connection	Address	Acquisition cycle
referencia1	Int	Modbus_Pixtend	4x400002	500 ms
referencia2	Int	Modbus_Pixtend	4x400003	500 ms
swControlador	Int	Modbus_Pixtend	4x400004	500 ms
nivel1	Int	Modbus_Pixtend	4x400005	500 ms
nivel2	Int	Modbus_Pixtend	4x400006	500 ms
sControl1	Int	Modbus_Pixtend	4x400007	500 ms
sControl2	Int	Modbus_Pixtend	4x400008	500 ms
sControl3	Int	Modbus_Pixtend	4x400009	500 ms
nivelEstimado1	Int	Modbus_Pixtend	4x400010	500 ms
nivelEstimado2	Int	Modbus_Pixtend	4x400011	500 ms
flujoEntrada	Int	Modbus_Pixtend	4x400012	500 ms
flujoSalida	Int	Modbus_Pixtend	4x400013	500 ms
swBomba	Int	Modbus_Pixtend	4x400014	500 ms
swModo	Int	Modbus_Pixtend	4x400015	500 ms
residuoEstimacion1	Int	Modbus_Pixtend	4x400016	500 ms
residuoEstimacion2	Int	Modbus_Pixtend	4x400017	500 ms
indicadorFuga1	Int	Modbus_Pixtend	4x400018	500 ms
indicadorFuga2	Int	Modbus_Pixtend	4x400019	500 ms

Figura 4.19: Configuración de las variables de proceso en el HMI

4.7. Calibración de los instrumentos de medición y actuadores

4.7.1. Calibración de los sensores de nivel

Las vibraciones típicas de la operación de la planta, las ondulaciones y salpicaduras del agua almacenada debido al caudal de ingreso a los tanques son factores que provocan la presencia de ruido en las mediciones. Esto debido principalmente al desvió y la obstaculización la trayectoria de las señales ultrasónicas que utiliza el módulo HC-SR04 para la medición de la distancia.

Para contrarrestar las ondulaciones y las salpicaduras de agua con el fin de reducir el ruido en las mediciones de nivel, se diseñó un flotador de madera para cada tanque, los flotadores tienen la finalidad de reducir el efecto de las ondulaciones del agua. Adicionalmente cada flotador ha sido provisto con un agujero que incluye una malla que amortigua el golpe del caudal que cae en los tanques, esto con el fin de reducir las salpicaduras de agua. En la Figura 4.20 se presentan los flotadores descritos.

A pesar de contrarrestar considerablemente los factores físicos que inducen ruido en las mediciones del nivel, fue necesario implementar adicionalmente las etapas de preprocesamiento de señales mostradas en la Figura 4.21 para disminuir el ruido en las mediciones. Como primera etapa, mediante la librería *NewPing* de *Arduino* se realiza un sobremuestreo de 4 mediciones del nivel actual de cada tanque. Estas mediciones son en primer lugar promediadas, con el fin de tener una estimación más confiable del nivel actual de cada tanque. Como segunda etapa, se utiliza un filtro de media móvil ponderada exponencialmente (EWMA, por sus siglas en inglés) para suavizar la señal medida del nivel en cada tanque. El filtro EWMA es definido por la Ecuación 4.3.

$$y_k = \alpha x_k + (1 - \alpha) y_{k-1} \tag{4.3}$$

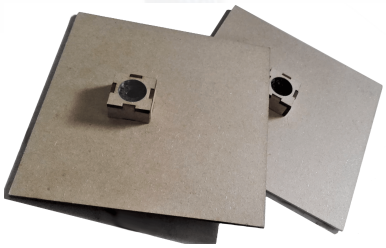


Figura 4.20: Flotadores para contrarrestar ondulaciones y salpicaduras

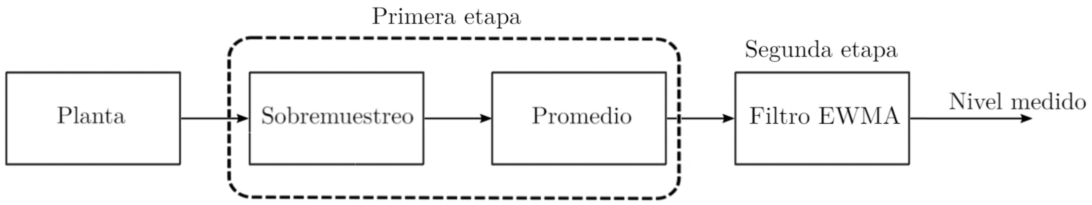


Figura 4.21: Etapas del preprocesamiento para la medición de nivel

Donde y_k es la muestra filtrada en el instante k, x_k es la muestra medida en el instante k y α es el valor de ajuste del suavizado de la señal. A un menor valor de α se pondera más los valores históricos de la señal, por lo cual presenta un comportamiento más suavizado. Si el valor de α es muy bajo se puede llegar a perder la capacidad de seguir la tendencia de la señal. El valor utilizado para α es de 0.5 y fue obtenido mediante pruebas experimentales para reducir el error de las mediciones y garantizar el funcionamiento del detector de fugas.

4.7.2. Calibración de las válvulas

Las señales generadas por los controladores comandan cada una de las válvulas para ejecutar las acciones de control. Sin embargo, la respuesta de apertura de las válvulas no tiene un comportamiento lineal, por lo cual se requirió determinar la función que relaciona la señal de control que comanda a la válvula con su porcentaje de apertura. Mediante mediciones se obtuvieron los datos que relacionan la señal de control con el porcentaje de apertura de las válvulas. Estos datos se aproximan a una función matemática mediante la herramienta *Curve Fitting Toolbox* de *MATLAB*.

4.7.2.1. Válvula Winner WVA4-3

La válvula proporcional Winner WVA4-3 en conjunto con la válvula de alivio de presión regulan el caudal de ingreso al sistema. Para determinar su función se midió el flujo de salida de la válvula para diferentes señales de control. Las mediciones realizadas se resumen en la Tabla 4.5, donde se presenta la señal de control ingresada a la válvula, su flujo de salida y el porcentaje de apertura equivalente a dicho flujo.

La función propuesta para la válvula se presenta en la Ecuación 4.4. En la Figura 4.22 se compara el comportamiento de la función con los valores medidos.

Tabla 4.5: Mediciones de la válvula Winner WVA4-3

Señal de control	Flujo (lt/min)	Porcentaje de apertura
0.050	0	0
0.100	0.0933	0.0231
0.125	0.1022	0.0253
0.150	0.2800	0.0693
0.175	0.9466	0.2343
0.200	2.3555	0.5830
0.225	3.8755	0.9593
0.250	4.0400	1

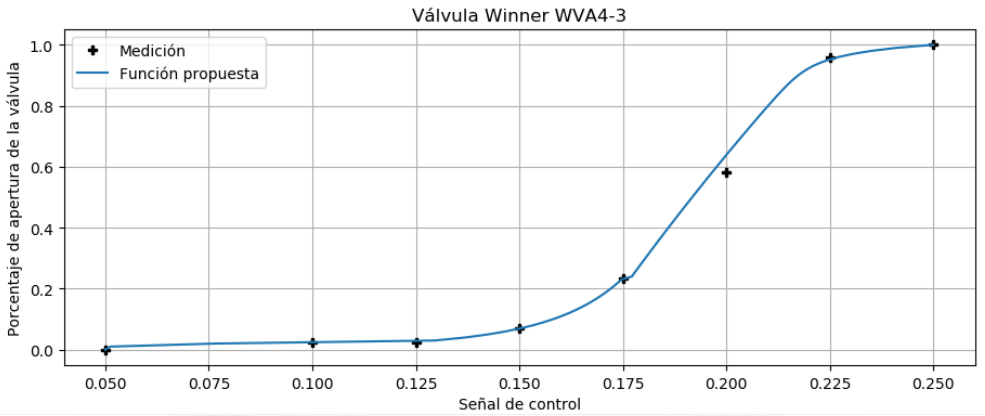


Figura 4.22: Función de transferencia propuesta para la válvula Winner WVA4-3

$$f(x) = \begin{cases} 9.578 \times 10^5 x^{4.45} + 0.05 & ,0 \leq x \leq 0.0253 \\ -0.08032x^{-0.1767} + 0.2788 & ,0.0253 < x \leq 0.2343 \\ 0.1647e^{0.3041x} + 2.551 \times 10^{-17} e^{34.57x} & ,0.2343 < x \leq 1 \end{cases} \tag{4.4}$$

4.7.2.2. Válvula BACOENG 2W-15

Para determinar experimentalmente la función de la válvula BACOENG 2W-15 se midió el tiempo que tomó vaciar el tanque desde una altura de 21.2 cm hasta 16.2 cm con diferentes señales de control. La Ecuación 4.5 presenta la relación entre el tiempo de vaciado y el área de la válvula.

$$a = \frac{A}{t}\sqrt{\frac{2}{g}}\left(\sqrt{h_1} - \sqrt{h_2}\right) \tag{4.5}$$

Donde a es el área transversal de la válvula, t es el tiempo de vaciado, h_1, h_2 son las alturas de referencia para la medición, g es la aceleración de la gravedad y A es el área transversal del tanque.

Los valores de áreas calculadas permiten establecer el porcentaje de apertura de la válvula para las diferentes señales de control. En la Tabla 4.6 se presenta un resumen de los tiempos medidos, las áreas calculadas y el porcentaje de apertura para diferentes señales de control ingresadas en la válvula.

Tabla 4.6: Mediciones de la válvula BACOENG 2W-15

Señal de control	**Tiempo de vaciado (s)**	**Área de salida (m^2)**	**Porcentaje de apertura**
0.050	-	0	0
0.100	236.62	0.442×10^{-5}	0.1010
0.140	89.42	1.170×10^{-5}	0.2672
0.150	76.71	1.364×10^{-5}	0.3114
0.155	62.31	1.679×10^{-5}	0.3834
0.160	55.68	1.879×10^{-5}	0.4291
0.170	45.89	2.280×10^{-5}	0.5206
0.180	42.97	2.435×10^{-5}	0.5560
0.190	39.76	2.632×10^{-5}	0.6069
0.200	38.31	2.732×10^{-5}	0.6236
0.225	26.43	3.959×10^{-5}	0.9039
0.250	23.89	4.380×10^{-5}	1

La función propuesta para la válvula se presenta en la Ecuación 4.6. En la Figura 4.23 se compara el comportamiento de la función con los valores medidos.

$$f(x) = 2.189x^5 - 5.942x^4 + 6.022x^3 - 2.835x^2 + 0.7665x + 0.0489 \tag{4.6}$$

4.7.2.3. Válvula U.S. SOLID USS-MSV00002

Para determinar la función de la válvula U.S. SOLID USS-MSV00002 se midió el tiempo que tomó vaciar el tanque desde una altura de 16.3 cm hasta 14.3 cm con diferentes señales de control. El área de la válvula se obtuvo por medio de la Ecuación 4.5. Los valores de áreas calculadas permiten establecer el porcentaje de apertura de la válvula para las diferentes señales de control. En la Tabla 4.7 se presenta un resumen de los tiempos medidos, las áreas calculadas y el porcentaje de apertura para diferentes señales de control ingresadas en la válvula.

La función propuesta para la válvula se presenta en la Ecuación 4.7 y en la Figura 4.24 se

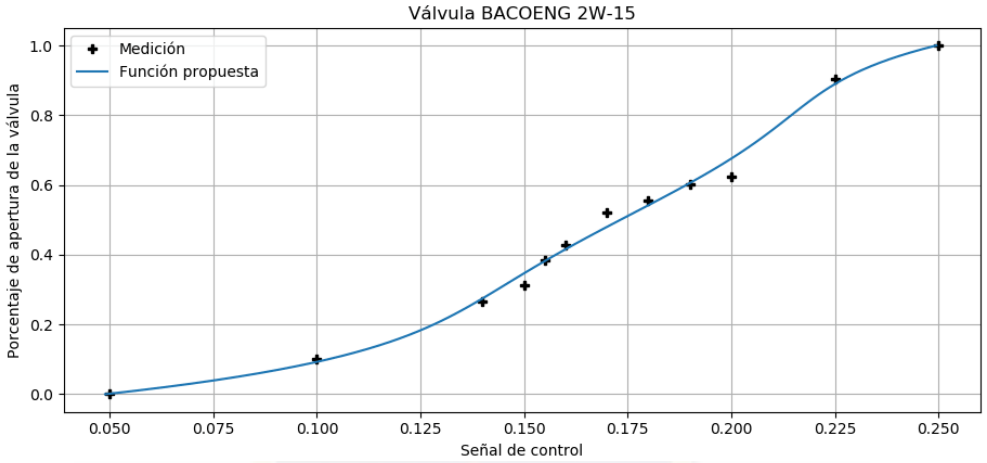


Figura 4.23: Función de transferencia propuesta para la válvula BACOENG 2W-15

Tabla 4.7: Mediciones de la válvula U.S. SOLID USS-MSV00002

Señal de control	**Tiempo de vaciado (s)**	**Área de salida (m^2)**	**Porcentaje de apertura**
0.10	-	0	0
0.20	41.06	1.125×10^{-5}	0.2445
0.25	20.87	2.214×10^{-5}	0.4811
0.35	14.72	3.138×10^{-5}	0.6821
0.50	11.93	3.872×10^{-5}	0.8416
0.60	11.21	4.121×10^{-5}	0.8956
0.75	10.56	4.375×10^{-5}	0.9508
0.90	10.05	4.597×10^{-5}	0.9990
1.00	10.04	4.601×10^{-5}	1

compara el comportamiento de la función con los valores medidos.

$$f(x) = 12.35x^5 - 28.71x^4 + 25.11x^3 - 9.510x^2 + 1.609x + 0.09998 \tag{4.7}$$

4.8. Modelado de la planta

A partir del diagrama P&ID mostrado en la Figura 4.2, se determina que el sistema posee dos variables de salida y tres variables de control. Las salidas del sistema corresponden a los niveles de agua de cada tanque, mientras que las variables de control representan el porcentaje de apertura de las válvulas. Estas ultimas se encargan de regular el flujo de entrada al Tanque 1, el flujo de salida del Tanque 1 que ingresa al Tanque 2 y el flujo de salida del Tanque 2

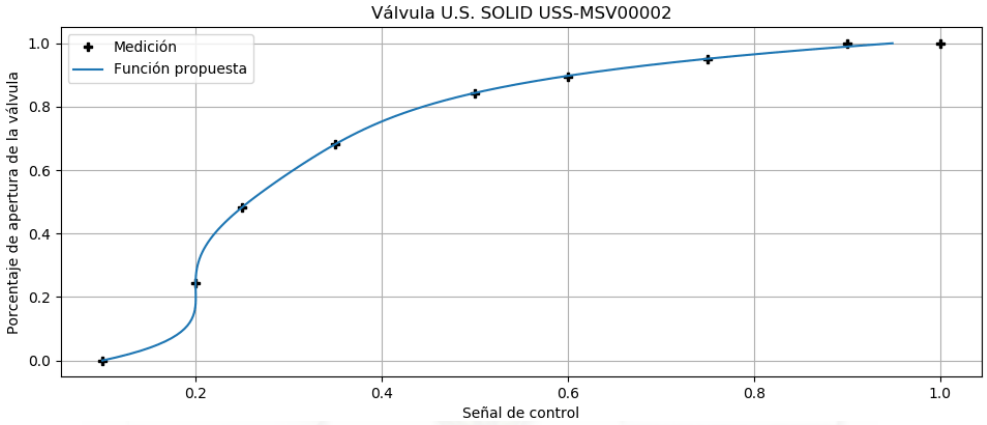


Figura 4.24: Función de transferencia propuesta para la válvula U.S. SOLID USS-MSV00002

respectivamente.

Las ecuaciones diferenciales que describen el sistema son las siguientes (Ecuación 4.8):

$$\begin{aligned} \frac{dh_1}{dt} &= \frac{1}{A_1}\left(u_1 q_0 - u_2 a_2 \sqrt{2gh_1}\right) \\ \frac{dh_2}{dt} &= \frac{1}{A_2}\left(u_2 a_2 \sqrt{2gh_1} - u_3 a_3 \sqrt{2gh_2}\right) \end{aligned} \tag{4.8}$$

Donde:

- h_1, h_2: Nivel de agua de los tanques 1 y 2.
- u_1, u_2, u_3: Porcentaje de apertura de las válvulas 1, 2 y 3 ($0 \leq u_{1,2,3} \leq 1$).
- A_1, A_2: Área de la sección transversal de los tanques 1 y 2.
- a_2, a_3: Área de la sección transversal de los tubos conectados a las válvulas 2 y 3.
- q_0: Caudal nominal de la bomba ($q_0 = 4$ lt/min).
- g: Aceleración de la gravedad.

Los valores considerados para las constantes físicas del sistema son los presentados en la Tabla 4.8.

Dentro del sistema se propone el modelado de fugas en los tanques para simular posibles fallas en la planta. Las fugas se modelan como un flujo de salida provocado por un agujero ubicado en el fondo del tanque. El modelo que incluye las fugas en los tanques es el siguiente (Ecuación 4.9):

Tabla 4.8: Constantes físicas del sistema

Parámetro	Valor
a_2	4.380×10^{-5} m^2
a_3	4.601×10^{-5} m^2
A_1	0.04 m^2
A_2	0.04 m^2
q_0	6.667×10^{-5} m^3/s
g	9.81 m/s^2

$$
\begin{aligned}
\frac{dh_1}{dt} &= \frac{1}{A_1}\left(u_1 q_0 - u_2 a_2 \sqrt{2gh_1} - l_1\right) \\
\frac{dh_2}{dt} &= \frac{1}{A_2}\left(u_2 a_2 \sqrt{2gh_1} - u_3 a_3 \sqrt{2gh_2} - l_2\right) \\
l_1 &= k_{l1} a_{l1} \sqrt{2gh_1} \\
l_2 &= k_{l2} a_{l2} \sqrt{2gh_2} \\
k_{l1,2} &= \begin{cases} 0, & \text{no existe fuga} \\ 1, & \text{existe fuga} \end{cases}
\end{aligned}
\tag{4.9}
$$

Donde:

- l_1, l_2: Flujo de salida debido a fugas en los tanques 1 y 2.
- a_{l1}, a_{l2}: Área de los agujeros que provocan la fuga en los tanques 1 y 2.
- k_{l1}, k_{l2}: Indicador si existe o no fuga en los tanques 1 y 2;

4.9. Desarrollo del sistema de control

El sistema de control planteado es de tipo descentralizado, es decir, se utiliza un controlador diferente por cada variable de control (Figura 4.25). Cada tanque presenta una referencia del nivel deseado (ref_{h_1} y ref_{h_2}). Las señales de error e_1 y e_3 se obtienen comparando los valores de referencia con los niveles medidos de los tanques (h_1 y h_2). La señal de error e_2 se obtiene comparando la diferencia de referencias ($ref_{h_1} - ref_{h_2}$) con la diferencia de alturas medidas ($h_1 - h_2$). Las señales de error ingresan en los controladores para obtener las señales de control u_1, u_2 y u_3, las cuales son enviadas a las válvulas para ejecutar acciones sobre la planta. Los controladores propuestos son digitales de tipo PID y FGS-PID. Finalmente se propone un observador implementado mediante un EKF que recibe como entrada los niveles de los tanques y las señales de control para realizar una estimación de los niveles de los tanques ($\tilde{h}_1$ y $\tilde{h}_2$),

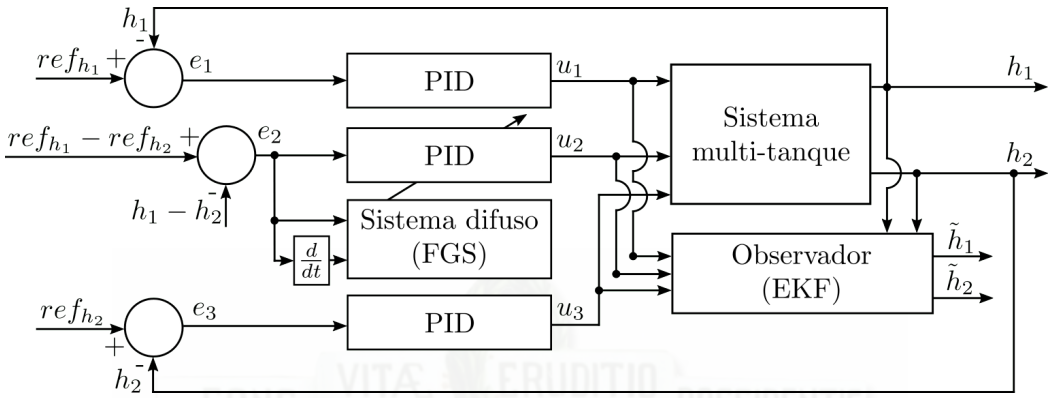


Figura 4.25: Sistema de Control planteado

usadas para la detección de fallas en el sistema.

En este trabajo se plantean dos esquemas de control. El esquema PID que se conforma un controlador PID por cada válvula y el esquema FGS-PID que posee controladores PID para las Válvulas 1 y 3 y un controlador FGS-PID para la Válvula 2. Se plantea el uso de un único controlador FGS-PID debido a que su tiempo de procesamiento llega a superar los 200 ms, por lo que implementar una mayor cantidad de controladores de este tipo puede producir desincronización en el periodo de muestreo del sistema debido al procesamiento realizado por los controladores. El controlador FGS-PID fue colocado en la Válvula 2 debido a que esta permite el control del nivel de los dos tanques y su tiempo de respuesta es menor que las otras dos válvulas, por lo cual responde de mejor manera a cambios en su señal de control.

4.9.1. Control PID

El sistema de control planteado utiliza un controlador PID por cada válvula en el sistema. Cada una de las válvulas posee un tiempo de respuesta diferente ante cambios en las señales de control, por lo cual es necesario considerar dicho tiempo en el desarrollo de los controladores.

La Ecuación 4.10 presenta el algoritmo de control discreto del PID. Para generar la señal de control u_k, el controlador requiere la señal de control previa u_{k-1}, el error actual y dos errores previos representados por los términos e_k, e_{k-1} y e_{k-2}. La sintonización del controlador se realiza por medio de los parámetros K_p, K_i y K_d, los cuales representan la parte proporcional, integral y derivativa, respectivamente.

$$u_k = u_{k-1} + K_p \left[e_k - e_{k-1} + K_i e_k + K_d \left(e_k - 2e_{k-1} + e_{k-2} \right) \right] \tag{4.10}$$

La sintonización de parámetros de los controladores se realizó con el objetivo de reducir el tiempo de estabilización y el porcentaje de sobreimpulso en el sistema, considerando la velocidad de reacción de las válvulas. Para la sintonización de los parámetros de los controladores se partió del método de ganancia crítica de *Ziegler-Nichols* en lazo cerrado y posteriormente se realizó un ajuste fino de los parámetros para obtener la respuesta deseada. Los resultados de la sintonización de los parámetros de cada controlador se muestran en la Tabla 4.9.

Tabla 4.9: Parámetros de los controladores PID

Parámetro	PID Válvula 1	PID Válvula 2	PID Válvula 3
K_p	1	2	5
K_i	2	-2	-5
K_d	4.5	-0.05	-0.05

4.9.2. Control PID con ganancia programada difusa (FGS-PID)

El controlador de tipo FGS-PID se implementó en la Válvula 2 que controla el flujo de salida del primer tanque. El esquema del control FGS-PID implementado se muestra en la Figura 4.26. El esquema es similar al descrito en la Sección 2.6.2 y se utilizan los mismos conjuntos de reglas difusas para el cálculo de los parámetros K_p, K_d y K_i, mientras que los conjuntos de funciones de membresía son ajustados al comportamiento de la planta.

Para las variables de entrada error $e(k)$ y derivada de error $\dot{e}(k)$ se determinaron siete funciones de membresía. Estos corresponden a errores negativos grandes(NB), negativos medios(NM), negativos pequeños(NS), cero(ZO), positivos pequeños(PS), positivos medios(PM) y positivos grandes(PB). Las funciones NM, NS, ZO, PS y PM son de tipo triangular, mientras que las funciones NB y PB son de tipo trapezoidal. En la Figura 4.27 se muestran las funciones de membresía implementadas para la variable error, mientras que la Figura 4.28 muestra las funciones de membresía determinadas para la variable derivada del error.

Las variables de salida $K_p{}'$ y $K_d{}'$ poseen dos funciones de membresía *Small* y *Big*, las cuales son de tipo Gaussianas con desviación estándar 0.4247 y centro en 0 y 1 respectivamente. Estas funciones de membresía se muestran en la Figura 4.29.

La variable de salida α posee cuatro funciones de membresía descritas como 2, 3, 4 y 5, las cuales son implementadas como funciones triangulares que simulan *singletons*. Estas funciones de membresía se muestran en la Figura 4.30.

Los valores límite para el cálculo de los parámetros K_p y K_d son los que se presentan en la Tabla 4.10.

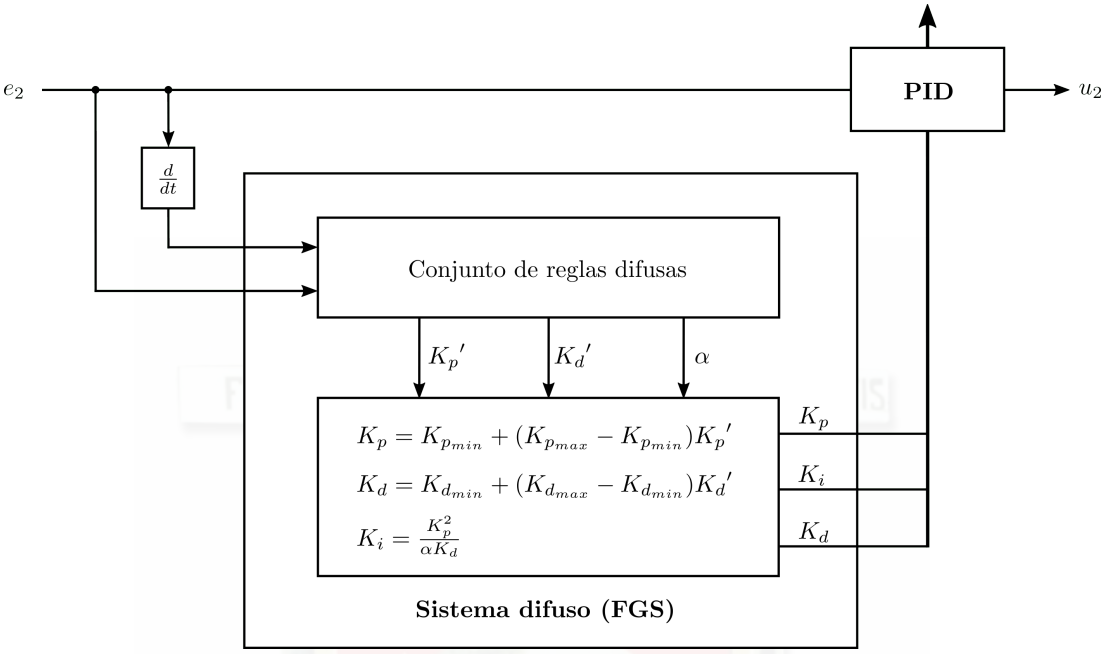


Figura 4.26: Esquema interno del controlador FGS-PID

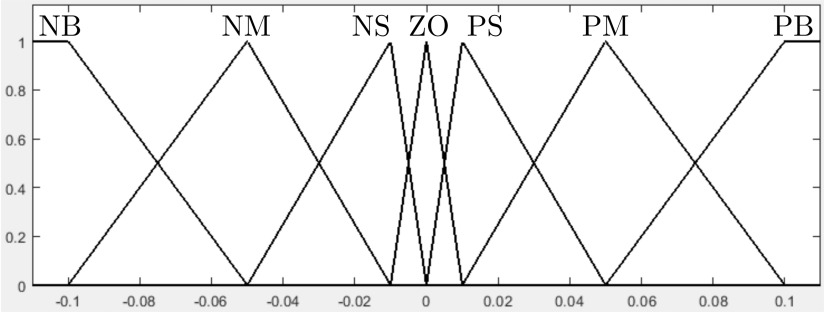


Figura 4.27: Funciones de membresía para la variable error

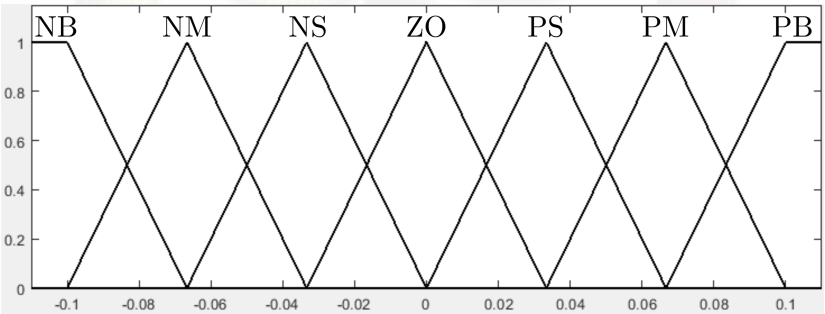


Figura 4.28: Funciones de membresía para la variable derivada del error

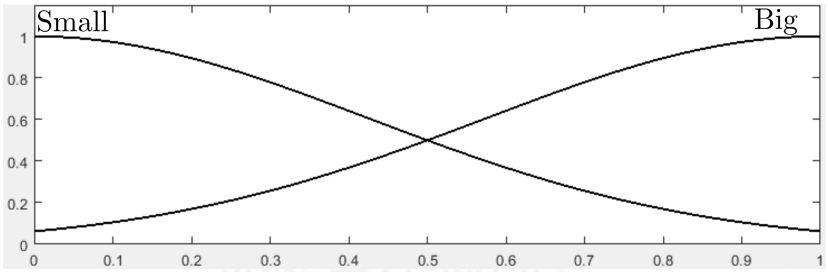


Figura 4.29: Funciones de membresía para las variables $K_p{}'$ y $K_d{}'$

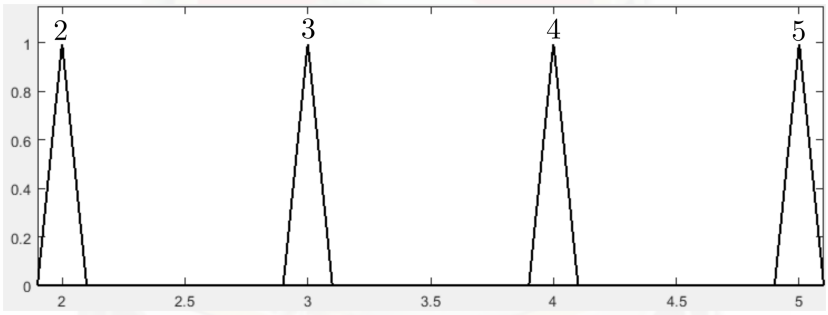


Figura 4.30: Funciones de membresía para la variable α

Tabla 4.10: Valores límite de los parámetros K_p y K_d

Parámetro	**Valor**
$K_{p_{\min}}$	12.5
$K_{p_{\max}}$	13.5
$K_{d_{\min}}$	-5.1
$K_{d_{\max}}$	-4.9

4.10. Sistema de detección de fallas

El sistema de detección de fallas tiene como objetivo notificar al operador la presencia de problemas dentro de la planta. En este trabajo se plantea la simulación de fugas en los tanques para evaluar el funcionamiento del sistema de detección de fallas. Este sistema se implementó mediante un EKF que actúa como observador. El EKF utiliza información de las señales de control y estimaciones previas del modelo de la planta para calcular el nivel actual de cada tanque. La estimación realizada por el EKF se compara con los niveles medidos y en base a su error es posible determinar la presencia de fallas en el sistema. Para desarrollar el EKF se requiere que la planta sea descrita por medio de un modelo lineal discreto en espacio de estados.

4.10.1. Linealización del sistema

El modelo del sistema multi-tanque presentado en la Ecuación 4.8 puede ser expresado como un sistema lineal en espacio de estado por medio de las relaciones de la Ecuación 4.11 [57].

$$\begin{aligned} \dot{x} &= Ax + Bu \\ y &= Cx + Du \\ x &= \begin{bmatrix} h_1 & h_2 \end{bmatrix}^T \\ u &= \begin{bmatrix} u_1 & u_2 & u_3 \end{bmatrix}^T \\ y &= \begin{bmatrix} h_1 & h_2 \end{bmatrix}^T \\ \dot{x} &= \frac{dx}{dt} \end{aligned} \tag{4.11}$$

Las matrices A, B, C y D que describen el modelo se obtienen linealizando el modelo mediante el método de linealización jacobiana. La linealización jacobiana es una representación matricial del término de primer orden de la expansión de la serie de Taylor de las ecuaciones diferenciales que modelan la planta. La linealización jacobiana del modelo se realiza evaluando el punto de equilibrio del sistema en un punto de operación. Los puntos de equilibrio se obtienen por medio de las relaciones descritas en la Ecuación 4.12 [58].

$$\begin{aligned} f_1 &= \frac{dh_1}{dt} = 0 \\ f_2 &= \frac{dh_2}{dt} = 0 \end{aligned} \tag{4.12}$$

Los puntos de equilibrio $x_e = \begin{bmatrix} h_{e_1} & h_{e_2} \end{bmatrix}^T$ se obtienen considerando que las señales de control

$u_e = [\ u_{e_1}\ \ u_{e_2}\ \ u_{e_3}]^T$ son el punto de operación de la planta y son valores conocidos. Las relaciones para obtener los puntos de equilibrio se presentan en la Ecuación 4.13.

$$\begin{aligned} h_{e_1} &= \frac{1}{2g}\left(\frac{q_0 u_{e_1}}{a_2 u_{e_2}}\right)^2 \\ h_{e_2} &= h_{e_1}\left(\frac{a_2 u_{e_2}}{a_3 u_{e_3}}\right)^2 \end{aligned} \tag{4.13}$$

En la Ecuación 4.14 se muestran las relaciones utilizadas para la linealización jacobiana [58].

$$\begin{aligned} A &= \left.\frac{df}{dx}\right|_{x=x_e,\ u=u_e} = \left.\begin{bmatrix} \frac{df_1}{dh_1} & \frac{df_1}{dh_2} \\ \frac{df_2}{dh_1} & \frac{df_2}{dh_2} \end{bmatrix}\right|_{x=x_e,\ u=u_e} \\ B &= \left.\frac{df}{du}\right|_{x=x_e,\ u=u_e} = \left.\begin{bmatrix} \frac{df_1}{du_1} & \frac{df_1}{du_2} & \frac{df_1}{du_3} \\ \frac{df_2}{du_1} & \frac{df_2}{du_2} & \frac{df_2}{du_3} \end{bmatrix}\right|_{x=x_e,\ u=u_e} \\ C &= \left.\frac{dy}{dx}\right|_{x=x_e,\ u=u_e} = \left.\begin{bmatrix} \frac{dy_1}{dh_1} & \frac{dy_1}{dh_2} \\ \frac{dy_2}{dh_1} & \frac{dy_2}{dh_2} \end{bmatrix}\right|_{x=x_e,\ u=u_e} \\ D &= \left.\frac{dy}{du}\right|_{x=x_e,\ u=u_e} = \left.\begin{bmatrix} \frac{dy_1}{du_1} & \frac{dy_1}{du_2} & \frac{dy_1}{du_3} \\ \frac{dy_2}{du_1} & \frac{dy_2}{du_2} & \frac{dy_2}{du_3} \end{bmatrix}\right|_{x=x_e,\ u=u_e} \end{aligned} \tag{4.14}$$

El resultado de la linealización se presenta en la Ecuación 4.15. Los valores de las diferentes constantes son los presentados en la Tabla 4.8 de la Sección 4.8.

$$\begin{aligned} A &= \begin{bmatrix} -\frac{a_2}{A_1}\sqrt{\frac{g}{2}}\frac{u_{e_2}}{\sqrt{h_{e_1}}} & 0 \\ \frac{a_2}{A_2}\sqrt{\frac{g}{2}}\frac{u_{e_2}}{\sqrt{h_{e_1}}} & -\frac{a_3}{A_2}\sqrt{\frac{g}{2}}\frac{u_{e_3}}{\sqrt{h_{e_2}}} \end{bmatrix} \\ B &= \begin{bmatrix} \frac{q_0}{A_1} & -\frac{a_2}{A_1}\sqrt{2gh_{e_1}} & 0 \\ 0 & \frac{a_2}{A_2}\sqrt{2gh_{e_1}} & -\frac{a_3}{A_2}\sqrt{2gh_{e_2}} \end{bmatrix} \\ C &= \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \\ D &= \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix} \end{aligned} \tag{4.15}$$

La discretización del modelo en espacio de estados para un periodo de muestreo T_s se obtiene por medio de las relaciones descritas en la Ecuación 4.16. El tiempo de muestro considerado para el sistema es de 500 ms.

$$
\begin{aligned}
A_d &= e^{AT_s} \\
B_d &= A^{-1}\left(A_d - I\right)B \\
C_d &= C \\
D_d &= D
\end{aligned}
\tag{4.16}
$$

4.10.2. Diseño del filtro extendido de Kalman (EKF)

El uso del EKF como un estimador de estados obedece a la naturaleza no lineal del sistema. Para la implementación del EKF se consideró el algoritmo presentado en la Sección 2.7.1, donde las matrices A y B cambian en cada instante de operación de la planta, debido a que la linealización del modelo se realiza para el estado actual del sistema.

Los parámetros modificables en el algoritmo del EKF son las matrices Q, R y P_k. Estos parámetros son los que inciden en la ganancia de corrección de la salida estimada con respecto a la medición de la salida del sistema. Estos parámetros son ajustados manualmente ya que no existe un método empírico para la obtención de los valores óptimos de dichos parámetros.

Como estado inicial del sistema se considera que $P_k = I$, donde I es una matriz identidad de 2×2. A la matriz Q se le asigna un valor de $1 \times 10^{-8} I$, mientras que para la matriz R se consideran dos escenarios. El primer escenario se presenta cuando inicia el sistema o existen cambios de referencia, en este caso se desea priorizar la etapa de corrección del EKF para realizar un seguimiento fiel de la señal medida. El valor seleccionado para este caso es $R = 1 \times 10^{-8} I$.

El segundo escenario se presenta cuando los niveles de los tanques ingresan en un rango de $\pm 5\,\%$ de error respecto a sus referencias, en este caso se prioriza la etapa de estimación del EKF seleccionando un valor $R = 1 \times 10^{-2} I$. Esta característica permite la detección de fallas utilizando el EKF como observador.

El EKF entrega como salidas, las estimaciones $\tilde{h}_1$ y $\tilde{h}_2$ de los niveles de los tanques. Las Ecuaciones 4.17 y 4.18 presentan los residuos obtenidos de comparar las estimaciones con los niveles medidos. Posteriormente, en las Secciones 5.1.3.1 y 5.2.3.1 se describe con detalle el uso del residuo para detectar fallas, tanto en la simulación como en la planta implementada físicamente.

$$r_1 = h_1 - \tilde{h}_1 \tag{4.17}$$

$$r_2 = h_2 - \tilde{h}_2 \tag{4.18}$$

4.11. Simulación del sistema multi-tanque

La simulación del sistema multi-tanque se desarrolló en *MATLAB* por medio de la herramienta *Simulink*. En la Figura 4.31 se presenta la implementación desarrollada en *Simulink*. El modelo se compone de los dos esquemas de control planteados funcionando en paralelo.

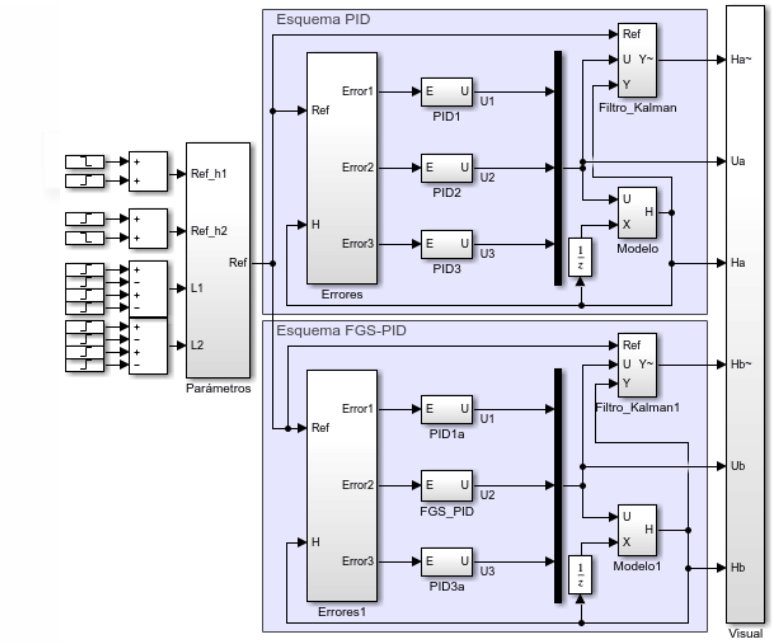


Figura 4.31: Implementación del sistema multi-tanque en *Simulink*

Cada esquema de control 6 diferentes subsistemas:

- **Parámetros:** Realiza la lectura de las referencias y fugas que ingresan al sistema. Este subsistema es común para los dos esquemas de control.
- **Cálculo de los errores:** Genera las señales de error por medio de la comparación de los niveles de los tanques con las referencias ingresadas al sistema.
- **Controladores:** Incluye los controladores que componen cada esquema de control.
- **Modelo de la planta:** Simula la planta en base a las ecuaciones diferenciales que describen el sistema.
- **Filtro de Kalman extendido:** Realiza la estimación de los niveles de los tanques para el proceso de detección de fallas en el sistema.
- **Visualización:** Permite observar los niveles de los tanques, referencias, residuos de las estimaciones, señales de control, fugas e índices de desempeño. Este subsistema es común para los dos esquemas de control.

El esquema PID consta de tres controladores tipo PID, uno por cada válvula. El esquema FGS-PID consta de un controlador tipo FGS-PID para la Válvula 2 y de controladores tipo PID para las válvulas 1 y 3. En la Figura 4.32 se muestra en detalle la implementación en *Simulink* del controlador FGS-PID. Este controlador se compone de un bloque *Fuzzy Logic Controller* encargado de ejecutar la lógica difusa para generar los parámetros que ingresan al PID.

Los controladores PID, el EKF, la simulación de fugas y el modelo de la planta son implementados por medio de *S-Functions*. El modelo se desarrolló por medio de la función *ode45* para resolver las ecuaciones diferenciales que describen la planta. Adicionalmente se incluye un bloque para generar ruido de medición.

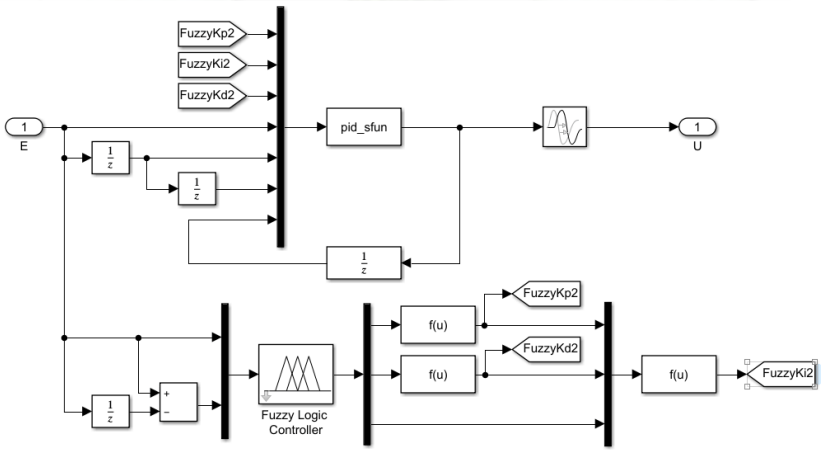


Figura 4.32: Implementación del controlador FGS-PID en *Simulink*

Capítulo 5

Pruebas de funcionamiento y evaluación de resultados

En este capítulo se comprueba el funcionamiento del sistema multi-tanque tanto a través de simulación como en su implementación real. Se evalúa el desempeño del sistema para el proceso de control, detección de fallas y comunicación con el servidor OPC-UA.

5.1. Resultados de simulación

El modelo presentado en la Sección 4.11 fue empleado para evaluar el desempeño de los sistemas de control de detección de fallas en MATLAB. El índice utilizado para evaluar los esquemas de control es el desempeño de seguimiento, descrito por la Ecuación 5.1. Este índice representa la capacidad del controlador de seguir una referencia. Un menor valor de este índice representa un mejor desempeño del controlador [59].

$$J = \sum_{k=1}^{+\infty} \left(\|h_1(k) - r_1(k)\|^2 + \|h_2(k) - r_2(k)\|^2 \right) \tag{5.1}$$

El sistema fue configurado de acuerdo a los parámetros físicos descritos en la Tabla 4.8. Adicionalmente, se consideró un periodo de muestreo de 500 ms, un radio de fuga de 1 cm y se incluyó ruido de medición con potencia 1×10^{-7}. Las pruebas de funcionamiento se realizaron para evaluar el desempeño tanto para el controlador tipo PID como del FGS-PID para controlar la Válvula 2.

5.1.1. Porcentaje de sobreimpulso y tiempo de estabilización en simulación

La primera prueba consistió en establecer un nivel de referencia de 12 cm para el Tanque 1 y de 8 cm para el Tanque 2, de esta manera se evaluó el sobreimpulso y tiempo de estabilización de los esquemas de control planteados.

En la Figura 5.1 se muestra la respuesta obtenida para los dos esquemas de control propuestos. Las señales en azul representan la respuesta al utilizar el controlador PID para la Válvula 2. En este caso se determinó que existe un sobreimpulso de 5.58 % para el Tanque 1 y de 8.21 % para el Tanque 2, mientras que el tiempo de estabilización es de 225 segundos. Las señales en rojo representan la respuesta al utilizar el controlador FGS-PID para la Válvula 2. En este caso se determinó que existe un sobreimpulso de 5.58 % para el Tanque 1 y de 7.65 % para el Tanque2, en tanto que el tiempo de estabilización es de 200 segundos.

En la Figura 5.2 se muestra una comparación del índice de desempeño de seguimiento para los dos tipos de controladores. Se puede observar que los dos esquemas de control presentan un desempeño similar. El índice de desempeño para el esquema PID es de 1.430, mientras que para el esquema FGS-PID es de 1.425. En este caso el uso del controlador FGS-PID permite una ligera mejora en el sistema de control.

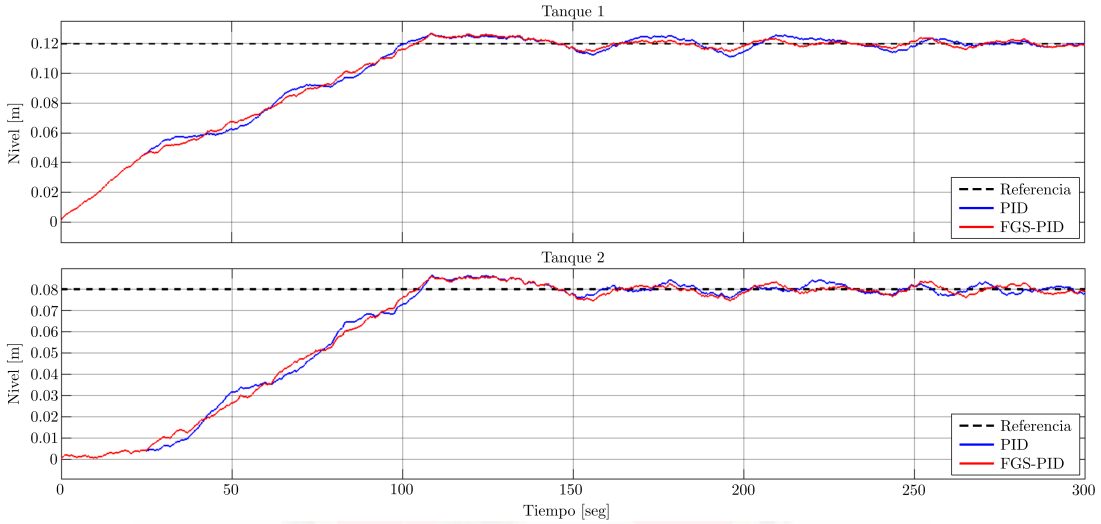


Figura 5.1: Respuesta en simulación de los esquemas de control planteados

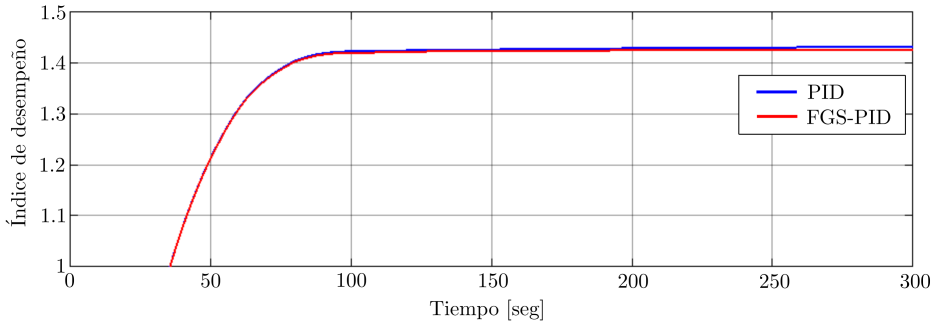


Figura 5.2: Índice de desempeño de los esquemas de control planteados

5.1.2. Cambios de referencia en simulación

La segunda prueba consistió en evaluar la respuesta del sistema ante cambios de referencia. En esta prueba, a los 200 segundos se realizó un cambio de referencia de 12 a 8 cm en el Tanque 1 y de 8 a 12 cm en el Tanque 2. A los 350 segundos se realizó un cambio de referencia de 8 a 12 cm en el Tanque 1. Finalmente, a los 400 segundos se realizó un cambio de referencia de 12 a 8 cm en el Tanque 2.

En la Figura 5.3 se muestra la respuesta obtenida al aplicar los cambios de referencia descritos anteriormente. Se puede observar que los controladores reaccionan correctamente para llevar el sistema a los nuevos valores deseados. Los dos esquemas de control planteados presentan resultados muy similares.

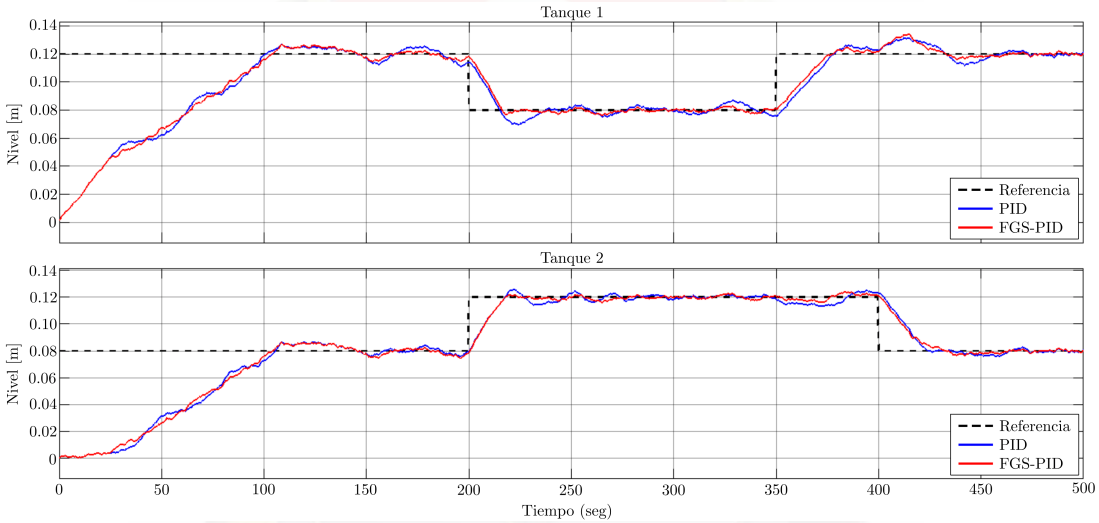


Figura 5.3: Respuesta en simulación de los esquemas de control planteados ante cambios de referencia

En la Figura 5.4 se muestra una comparación del índice de desempeño de seguimiento para los dos tipos de controladores. El índice de desempeño para el esquema PID es de 1.543, mientras que para el esquema FGS-PID es de 1.522. En este caso la mejora del sistema al utilizar el controlador FGS-PID es un poco más notoria.

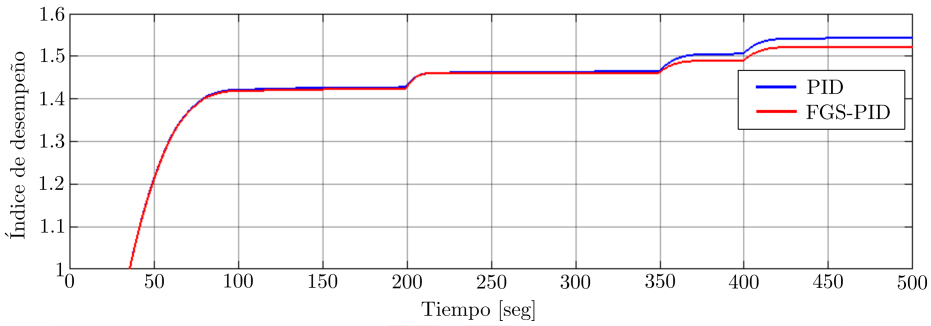


Figura 5.4: Índice de desempeño de los esquemas de control planteados ante cambios de referencia

5.1.3. Detección de fallas en simulación

5.1.3.1. Corrección del filtro de Kalman extendido (EKF) y reglas para la detección de fallas en simulación

Las matrices A y B correspondientes al sistema lineal en tiempo continuo descrito en la Ecuación 4.15 debieron ser ajustadas para obtener un desempeño correcto del EKF y detectar en qué tanque se encuentra una fuga. El ajuste se realiza por medio de matrices de ajuste A_{ajuste} y B_{ajuste}, las cuales son las presentadas en las Ecuaciones 5.2 y 5.3. Estas matrices de ajuste son empleadas tanto para el controlador PID como FGS-PID.

$$A_{\text{ajuste}} = \begin{bmatrix} 1 & 1 \\ 1 & 1 \end{bmatrix} \tag{5.2}$$

$$B_{\text{ajuste}} = \begin{bmatrix} 1 & 0.6 & 1 \\ 1 & 0.2 & 0.25 \end{bmatrix} \tag{5.3}$$

Las matrices $A_{\text{corregida}}$ y $B_{\text{corregida}}$ correspondientes se obtienen al realizar la multiplicación elemento a elemento de las matrices A y B por sus matrices de ajuste respectivas. Esto de acuerdo a lo mostrado en la Ecuación 5.4, donde $\cdot$ denota la multiplicación elemento a elemento entre las matrices.

$$\begin{aligned} A_{\text{corregida}} &= A \cdot A_{\text{ajuste}} \\ B_{\text{corregida}} &= B \cdot B_{\text{ajuste}} \end{aligned} \tag{5.4}$$

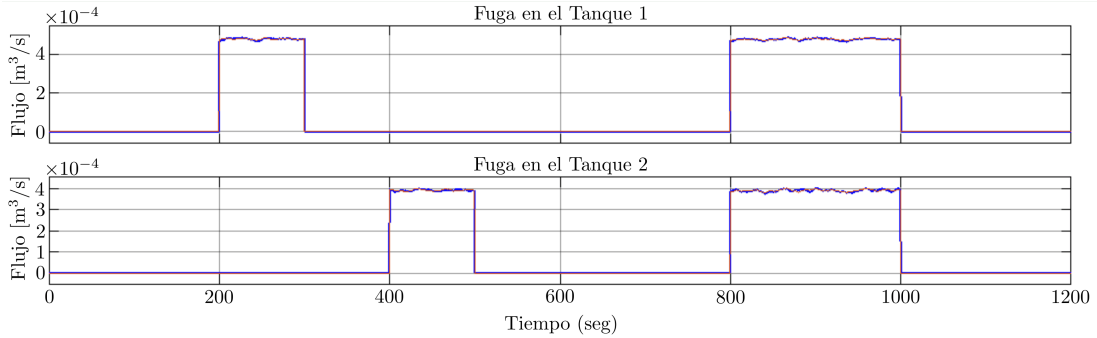


Figura 5.5: Patrón de fugas utilizado para la evaluación del sistema de detección de fallas en *Simulink*

Debido a que el EKF se diseñó en tiempo discreto, las matrices empleadas son las obtenidas en la discretización. Estas se calculan mediante las relaciones descritas en la Ecuación 5.5.

$$\begin{aligned} A_d &= e^{A_{\text{corregida}} T_s} \\ B_d &= A_{\text{corregida}}^{-1} \left(A_d - I\right) B_{\text{corregida}} \end{aligned} \tag{5.5}$$

Las reglas para detectar fallas en la simulación consisten en establecer valores umbrales para los residuos de las estimaciones, de manera que si el residuo es menor que el valor umbral se determina que existe una fuga en dicho tanque. El umbral establecido para los residuos es -0.01. Este umbral fue escogido en base a mediciones con el fin de evitar falsas alarmas de presencia de fugas.

5.1.3.2. Evaluación de la detección de fallas en simulación

Para evaluar el sistema de detección de fallas se estableció la referencia del Tanque 1 en 12 cm y la del Tanque 2 en 8 cm. Entre los 200 y 300 segundos se simuló una fuga en el Tanque 1. Entre los 400 y 500 segundos se simuló una fuga en el Tanque 2. Finalmente entre los 800 y 1000 segundos se simulan las fugas simultáneas en los dos tanques. La Figura 5.5 muestra el patrón de fugas utilizado para evaluar la detección de fallos.

Por su parte, en la Figura 5.6 se muestra el residuo de estimación obtenido con el EKF para los dos esquemas de control planteados. Las señales en azul representan los residuos al utilizar el controlador PID, las señales en rojo representan los residuos al utilizar el controlador FGS-PID y las señales en negro son los umbrales de detección de fugas. Se puede observar que en los

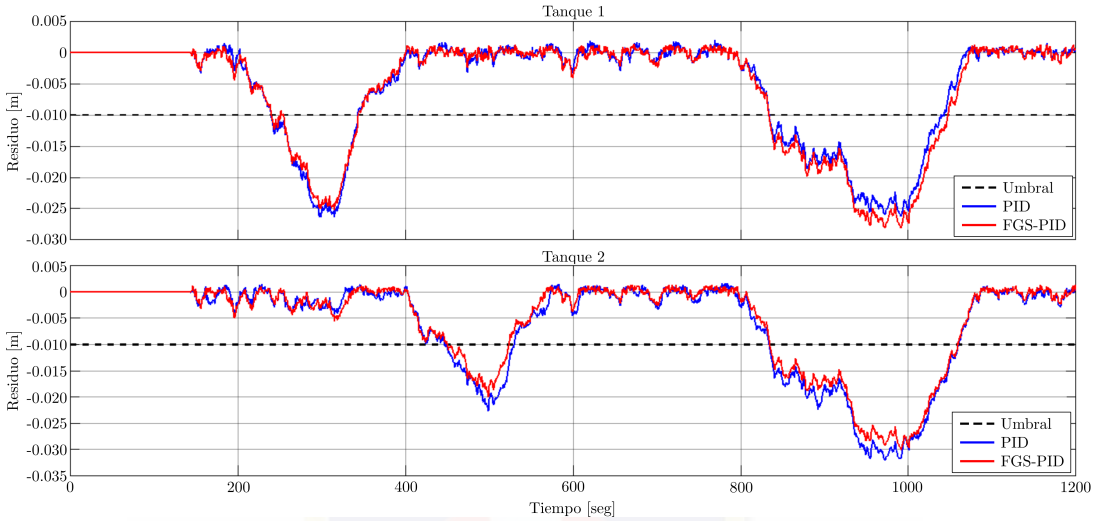


Figura 5.6: Respuesta en simulación del sistema de detección de fallas utilizando el controlador PID para la Válvula 2

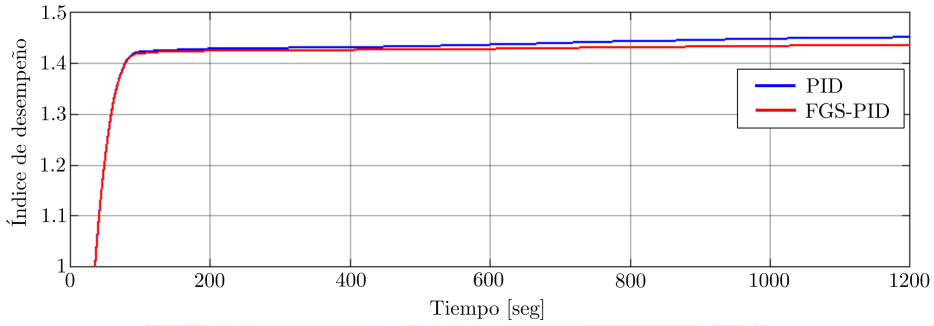


Figura 5.7: Índice de desempeño de los esquemas de control planteados ante fugas en el sistema

dos casos, los residuos de los tanques permiten detectar fugas cuando su valor es menor a sus respectivos umbrales. Al quitar las fugas los valores de residuo incrementan su valor y permiten determinar la ausencia de fugas cuando su valor es mayor a su umbral respectivo. Se puede observar que el EKF funciona de manera similar con los dos esquemas de control planteados.

En la Figura 5.7 se muestra una comparación del índice de desempeño de seguimiento para los dos tipos de controladores. El índice de desempeño para el esquema PID es de 1.451, mientras que para el esquema FGS-PID es de 1.436. De igual manera se puede observar que el uso del controlador FGS-PID permite un mejor desempeño del sistema.

En la Tabla 5.1 se presenta un resumen de los índices de desempeño obtenidos para las pruebas realizadas en la simulación. Se puede observar que en general el uso del controlador FGS-PID

permite mejorar el rendimiento del sistema de control.

Tabla 5.1: Tabla comparativa del índice de desempeño de los esquemas de control planteados

Prueba	**Índice de desempeño**	
	PID	FGS-PID
Estabilización en la referencia	1.430	1.425
Cambios de referencia	1.543	1.522
Fugas en el sistema	1.451	1.436

5.2. Funcionamiento de la implementación del sistema multi-tanque

En la Figura 5.8 se muestra el sistema multi-tanque construido y la instalación de los sensores y actuadores para el control de la planta. Por otro lado en la Figura 5.9 se muestra la instalación del tablero de instrumentación construido e instalado. La construcción y la instalación, tanto de la planta como del tablero, se realizaron en base al diseño descrito en la Sección 4.4.

5.2.1. Porcentaje de sobreimpulso y tiempo de estabilización en la planta

En la Figura 5.10 se muestra la respuesta de la planta al utilizar el controlador PID para la Válvula 2, en este caso se determinó que existe un sobreimpulso de 3 % para el Tanque 1 y de 8.75 % para el tanque2, mientras que el tiempo de estabilización es de 243 segundos.

En la Figura 5.11 se muestra la respuesta de la planta al utilizar el controlador FGS-PID para la Válvula 2, en este caso se determinó que existe un sobreimpulso de 3 % para el Tanque 1 y de 4.75 % para el Tanque 2, mientras que el tiempo de estabilización es de 212 segundos.

Comparando los resultados obtenidos se puede concluir que los resultados de la planta real son mejores a los obtenidos mediante simulación. Esto se debe principalmente a que los parámetros de los controladores, tanto de la simulación como de la implementación real, son los mismos y además fueron calibrados directamente en la planta real de manera experimental. A diferencia de los resultados de la simulación, el uso del controlador FGS-PID provoca que la respuesta del sistema tenga menor sobreimpulso y su estabilización se realiza en menor tiempo, además de que el nivel del Tanque 2 presenta oscilaciones menos pronunciadas.

Figura 5.8: Planta del sistema multi-tanque con los sensores y actuadores instalados

Figura 5.9: Tablero de instrumentación construido e instalado

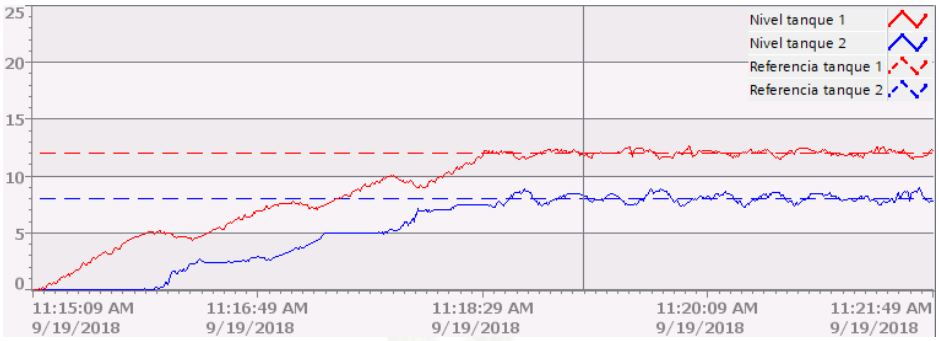


Figura 5.10: Respuesta de la planta utilizando el controlador PID para la Válvula 2

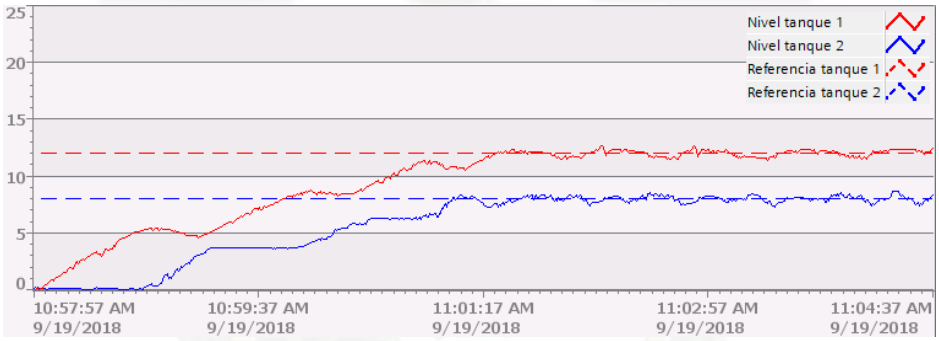


Figura 5.11: Respuesta de la planta utilizando el controlador FGS-PID para la Válvula 2

En la Figura 5.12 se muestra el comportamiento de la señal de control para la Válvula 2 cuando se utilizan los dos tipos de controladores propuestos. La primera parte de la señal en azul es la generada por el controlador FGS-PID. En este caso se observa una gran cantidad de variaciones, debido a los cambios en los parámetros del PID. La segunda parte de la señal en azul es la generada por el controlador PID clásico, la cual presenta un comportamiento más suavizado.

5.2.2. Cambios de referencia en la planta

En la Figura 5.13 se muestra la respuesta de la planta ante cambios de referencia al utilizar el controlador PID para la Válvula 2. Se puede observar que los controladores responden correctamente ante cambios de referencia de los dos tanques.

La Figura 5.13 también muestra la respuesta de la planta ante cambios de referencia al utilizar el controlador FGS-PID para la Válvula 2. Al igual que en el caso anterior, los controladores permiten estabilizar el sistema ante cambios de referencia de los dos tanques.

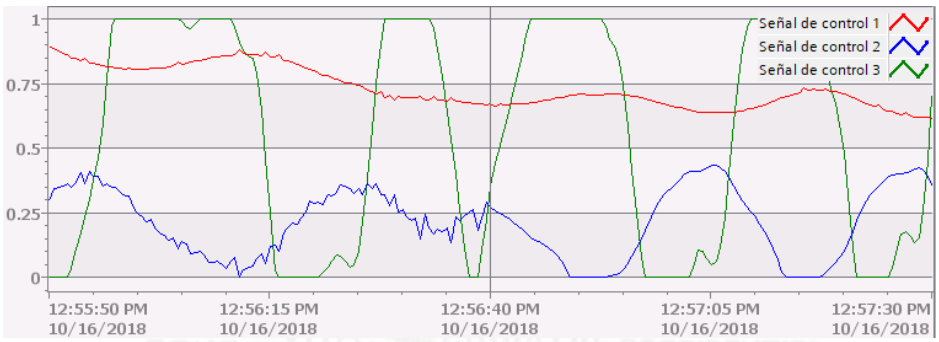


Figura 5.12: Comparación de la señal de control generada por el FGS-PID y por el PID

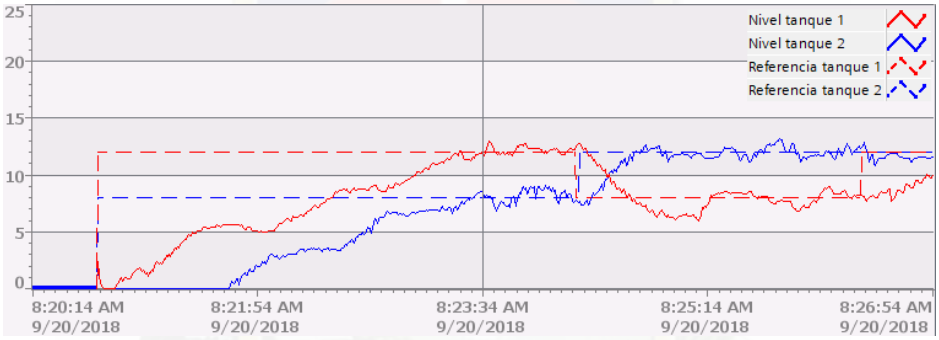


Figura 5.13: Respuesta de la planta ante cambios de referencia utilizando el controlador PID para la Válvula 2

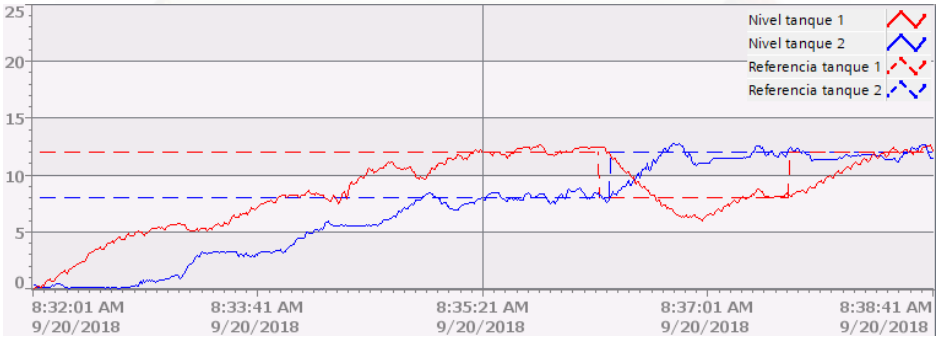


Figura 5.14: Respuesta de la planta ante cambios de referencia utilizando el controlador FGS-PID para la Válvula 2

5.2.3. Detección de fallas en la planta

5.2.3.1. Corrección del filtro de Kalman extendido (EKF) y reglas para la detección de fallas en la planta

De igual manera que lo expuesto en la Sección 5.1.3.1, se requirió utilizar matrices de ajuste para mejorar el desempeño del EKF en la planta. Estas matrices han sido obtenidas de manera experimental mediante el método de prueba y error. A diferencia de la simulación, en la planta se requirió utilizar matrices de ajuste diferentes para cada tipo de controlador, las cuales fueron obtenidas de manera experimental sobre la planta con el objetivo de aislar las fugas, de manera que sea posible detectar en que tanque se produce una fuga. En las Ecuaciones 5.6 y 5.7 se muestran las matrices de ajuste empleadas cuando se utiliza el controlador PID.

$$A_{\text{ajustePID}} = \begin{bmatrix} 2.2 & 1 \\ 1.05 & 1.05 \end{bmatrix} \tag{5.6}$$

$$B_{\text{ajustePID}} = \begin{bmatrix} 1 & 0.80 & 1 \\ 1 & 0.268 & 0.25 \end{bmatrix} \tag{5.7}$$

En las Ecuaciones 5.8 y 5.9 se muestran las matrices de ajuste empleadas cuando se utiliza el controlador FGS-PID.

$$A_{\text{ajusteFGSPID}} = \begin{bmatrix} 2.2 & 1 \\ 1.05 & 1.05 \end{bmatrix} \tag{5.8}$$

$$B_{\text{ajusteFGSPID}} = \begin{bmatrix} 0.52 & 0.40 & 1 \\ 1 & 0.222 & 0.188 \end{bmatrix} \tag{5.9}$$

Las reglas para detectar fallas en la planta, al igual que en la simulación, consisten en establecer valores umbrales para los residuos de las estimaciones. En el caso de la planta se manejan dos umbrales por cada residuo, el primer umbral se denomina de detección y permite establecer que existe una fuga, mientras que el segundo umbral se denomina de recuperación y se utiliza para determinar que ya no existe fuga en el sistema. El valor del umbral de detección se estableció en -0.015 para el esquema PID. Para el esquema FGS-PID se maneja un umbral de -0.010 cuando la referencia del Tanque 1 es mayor a la del Tanque 2 y un umbral de -0.015 para los otros casos. Estos valores se obtuvieron de manera experimental en base a mediciones para evitar falsos positivos en la detección de fugas. El valor del umbral de recuperación se estableció en 0

para los dos esquemas de control, este valor se determinó de manera experimental para evitar falsos negativos en la detección de fugas.

Cuando una señal de residuo mantiene durante 5 periodos continuos de muestreo un valor inferior al umbral de detección, se determina que existe una fuga en dicho tanque y se procede a activar su indicador de fuga. El indicador de fuga se desactiva cuando la señal de residuo mantiene durante 5 periodos de muestreo un valor superior al umbral de recuperación, lo cual permite determinar que ya no existe fuga en el tanque.

En las Figuras 5.15 - 5.30, presentadas en las Secciones 5.2.3.2 y 5.2.3.3, se muestra el comportamiento de los residuos de estimación. En dichas figuras la señal de residuo se encuentra en color rojo y se puede observar que ante fugas, las señales de residuo decrecen su valor. Por su parte, el indicador de fuga respectivo se activa cuando el valor de las señales de residuo es inferior al valor de umbral de detección definido. Al retirar las fugas, las señales de residuo aumentan su valor y el indicador de fuga respectivo se desactiva cuando el valor de las señales de residuo es superior al valor de umbral de recuperación definido.

5.2.3.2. Evaluación de la detección de fallas utilizando el controlador PID para la Válvula 2 en la planta

La primera prueba realizada consiste en introducir una fuga en el Tanque 1. En la Figura 5.15 se muestra la detección de fuga en el Tanque 1. La fuga fue detectada alrededor de 67 segundos después de ser colocada.

En la Figura 5.16 se muestra el comportamiento de la estimación del Tanque 1 luego de tapar la fuga. La desactivación del indicador de fuga se produjo alrededor de 116 segundos después de quitar la fuga.

La segunda prueba realizada consiste en introducir una fuga en el Tanque 2. En la Figura 5.17 se muestra la detección de una fuga en el Tanque 2. La fuga fue detectada alrededor de 245 segundos después de ser colocada.

En la Figura 5.18 se muestra el comportamiento de la estimación del Tanque 2 luego de tapar la fuga. La desactivación del indicador de fuga se produjo alrededor de 73 segundos después de quitar la fuga.

La última prueba realizada consiste en introducir una fuga en los dos tanques. En las figuras 5.19 y 5.20 se muestra el comportamiento de las estimaciones de los tanques 1 y 2 respectivamente. La fuga en el Tanque 1 fue detectada alrededor de 36 segundos después de ser colocada la fuga. La fuga en el Tanque 2 fue detectada después de 175 segundos de ser colocada la fuga.

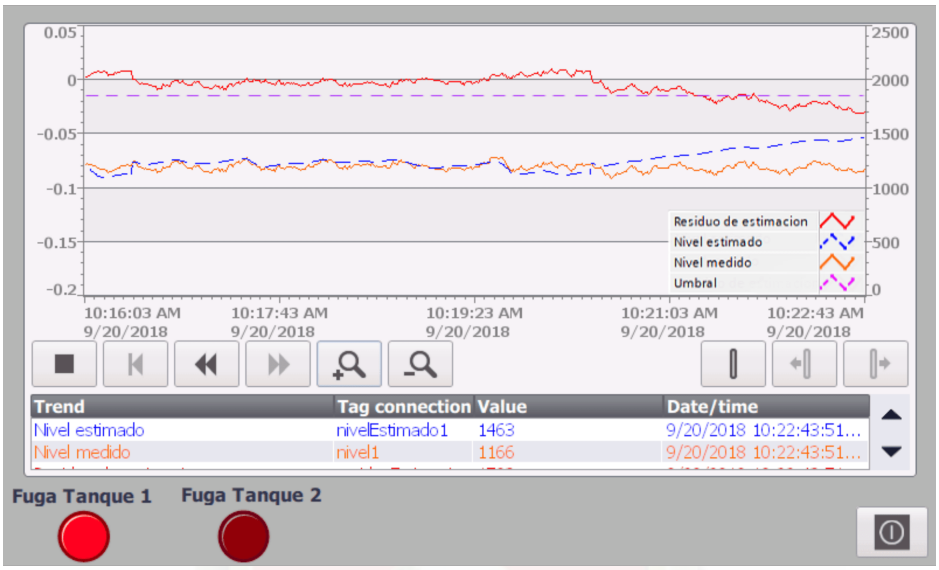


Figura 5.15: Detección de una fuga en el Tanque 1 en la planta utilizando el controlador PID para la Válvula 2

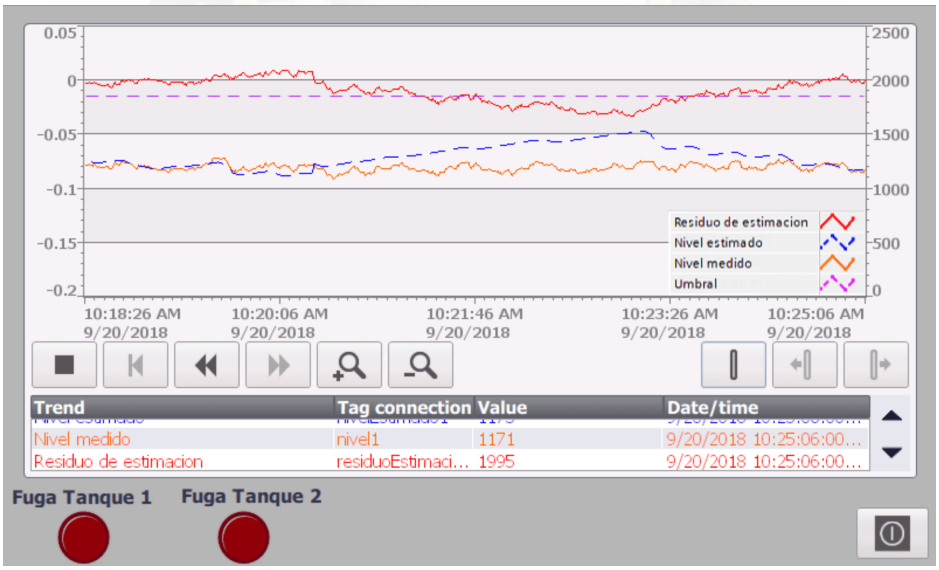


Figura 5.16: Recuperación de la estimación al quitar la fuga en el Tanque 1 en la planta utilizando el controlador PID para la Válvula 2

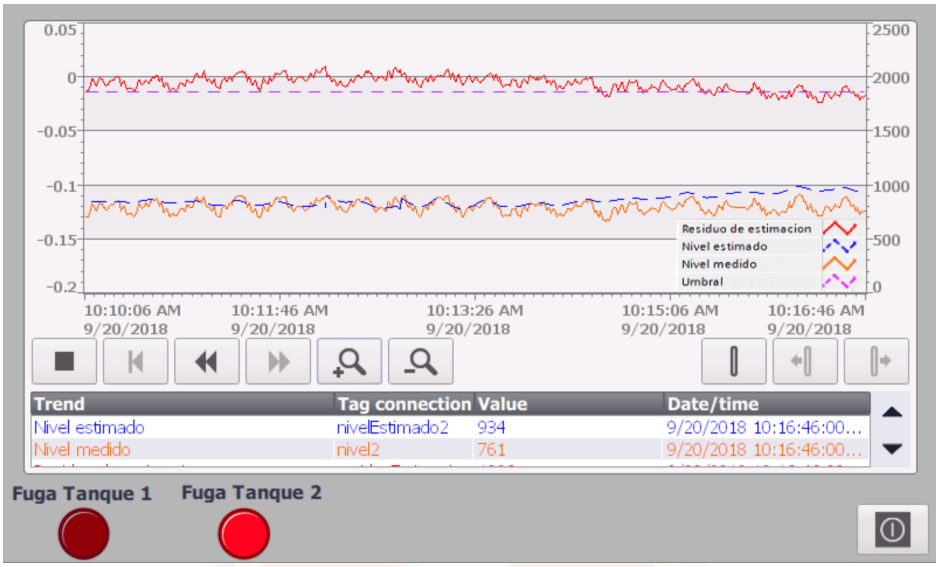


Figura 5.17: Detección de una fuga en el Tanque 2 en la planta utilizando el controlador PID para la Válvula 2

Figura 5.18: Recuperación de la estimación al quitar la fuga en el Tanque 2 en la planta utilizando el controlador PID para la Válvula 2

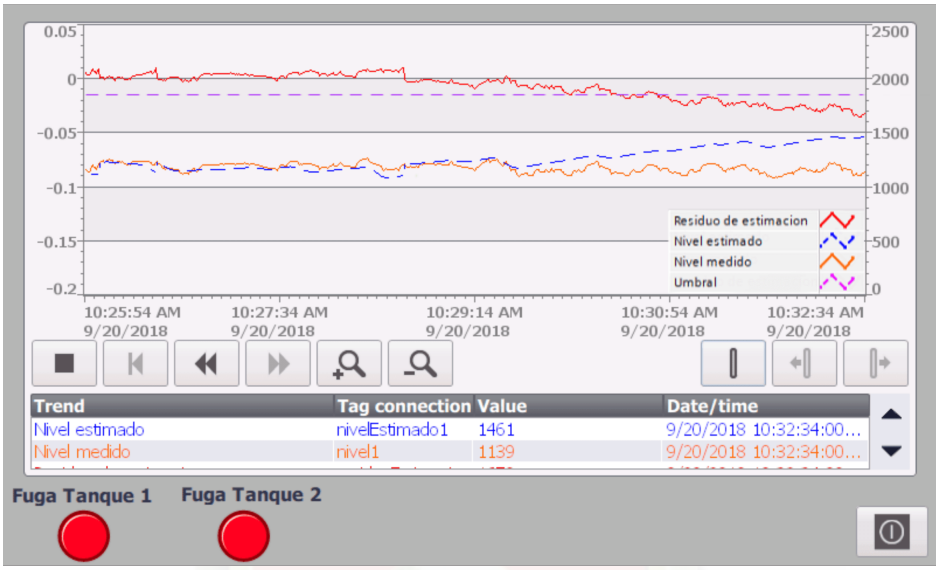


Figura 5.19: Detección de fuga en el Tanque 1 de la planta cuando existen fugas en los dos tanques utilizando el controlador PID para la Válvula 2

Figura 5.20: Detección de fuga en el Tanque 2 de la planta cuando existen fugas en los dos tanques utilizando el controlador PID para la Válvula 2

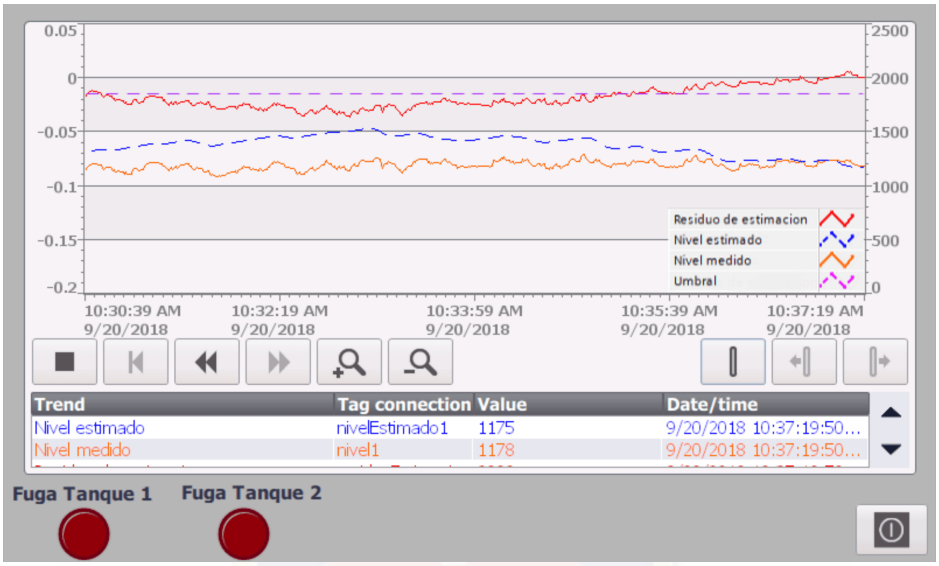


Figura 5.21: Recuperación de la estimación del Tanque 1 en la planta al quitar las fugas en los dos tanques utilizando el controlador PID para la Válvula 2

En las figuras 5.21 y 5.22 se muestra la recuperación de las estimaciones de los tanques 1 y 2 luego de quitar las fugas. El indicador de fuga en el Tanque 1 fue desactivado luego de 108 segundos de ser quitada la fuga. El indicador de fuga en el Tanque 2 fue desactivado luego de 63 segundos de ser quitada la fuga.

5.2.3.3. Evaluación de la detección de fallas utilizando el controlador FGS-PID para la Válvula 2 en la planta

El conjunto de pruebas realizadas al utilizar el controlador FGS-PID para la Válvula 2 es el mismo que el presentado en la Sección 5.2.3.2. En la Figura 5.23 se muestra la detección de una fuga en el Tanque 1. La fuga fue detectada alrededor de 124 segundos después de ser colocada.

En la Figura 5.24 se muestra el comportamiento de la estimación del Tanque 1 luego de tapar la fuga. La desactivación del indicador de fuga se produjo alrededor de 68 segundos después de quitar la fuga.

En la Figura 5.25 se muestra la detección de una fuga en el Tanque 2. La fuga fue detectada 156 segundos después de ser colocada.

En la Figura 5.26 se muestra el comportamiento de la estimación del Tanque 2 luego de tapar la fuga. La desactivación del indicador de fuga se produjo alrededor de 36 segundos después de quitar la fuga.

Figura 5.22: Recuperación de la estimación del Tanque 2 en la planta al quitar las fugas en los dos tanques utilizando el controlador PID para la Válvula 2

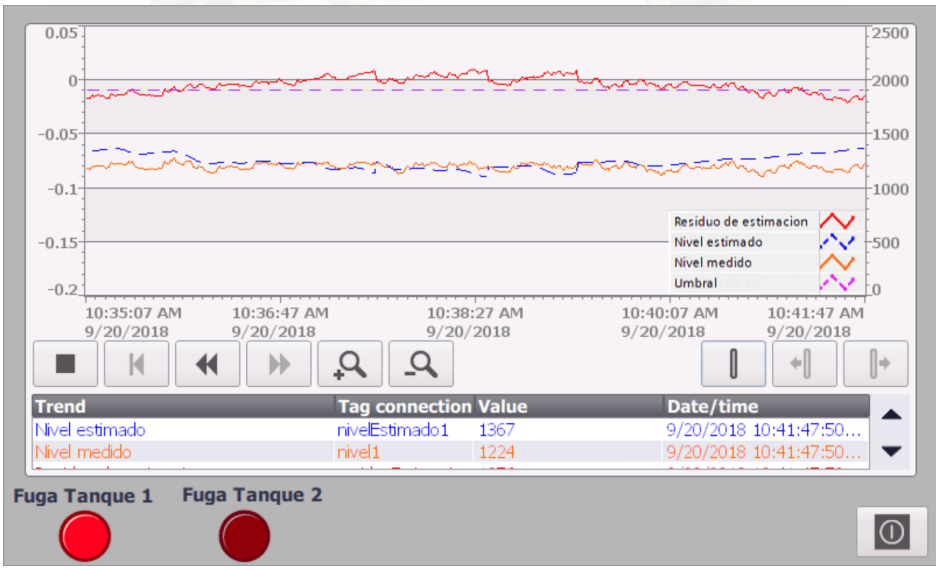


Figura 5.23: Detección de una fuga en el Tanque 1 en la planta utilizando el controlador FGS-PID para la Válvula 2

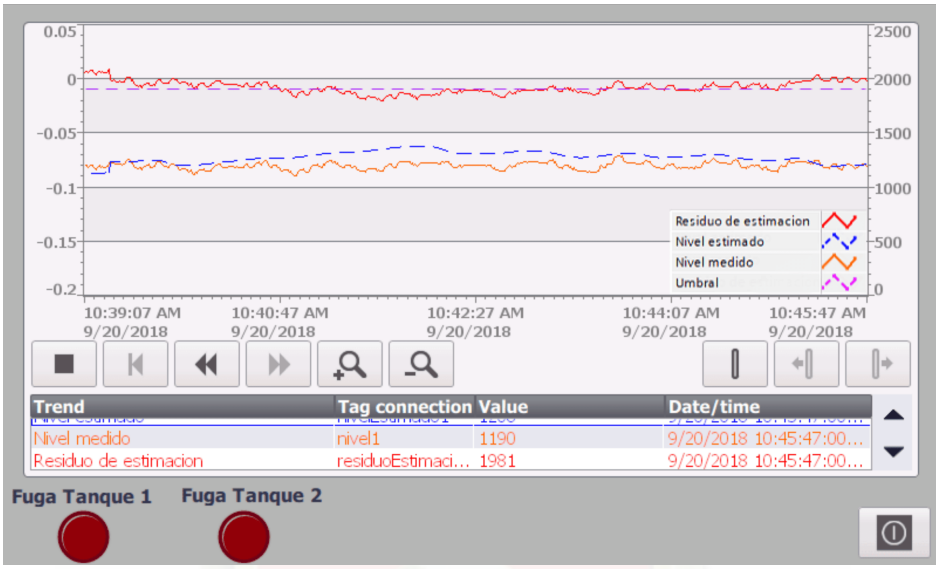


Figura 5.24: Recuperación de la estimación al quitar la fuga en el Tanque 1 en la planta utilizando el controlador FGS-PID para la Válvula 2

Figura 5.25: Detección de una fuga en el Tanque 2 en la planta utilizando el controlador FGS-PID para la Válvula 2

Figura 5.26: Recuperación de la estimación al quitar la fuga en el Tanque 2 en la planta utilizando el controlador FGS-PID para la Válvula 2

La última prueba realizada consiste en introducir una fuga en los dos tanques. En las Figuras 5.27 y 5.28 se muestra el comportamiento de las estimaciones de los tanques 1 y 2 respectivamente. La fuga en el Tanque 1 fue detectada alrededor de 105 segundos después de ser colocada. La fuga en el Tanque 2 fue detectada después de 117 segundos de ser colocada.

En las Figuras 5.21 y 5.22 se muestra la recuperación de las estimaciones de los tanques 1 y 2 luego de quitar las fugas. El indicador de fuga en el Tanque 1 fue desactivado luego de 202 segundos de ser quitada la fuga. El indicador de fuga en el Tanque 2 fue desactivado luego de 31 segundos de ser quitada la fuga.

En la Tabla 5.2 se presenta una comparación de los tiempos de activación y desactivación de alarmas de fuga para los dos esquemas de control planteados. Estos valores se obtuvieron realizando 5 mediciones de los tiempos de activación y desactivación de alarmas. A partir de estas mediciones, se calcula su promedio y se determina su intervalo de confianza de 95 %. En dichos resultados se observa que el intervalo de confianza de los resultados es amplio. Esto se debe a que el modelo lineal que utiliza el observador de estados es muy dependiente del valor de los niveles y de las señales de control en el sistema.

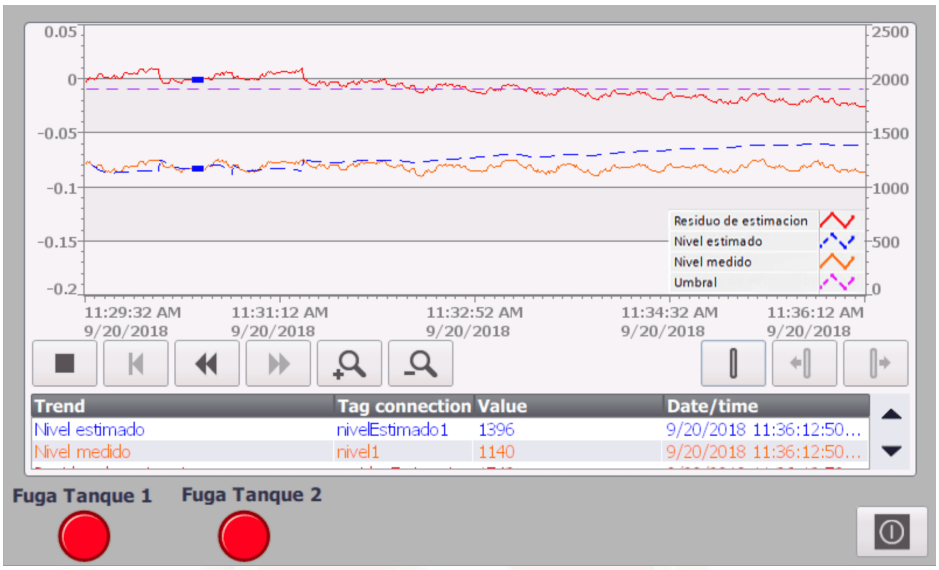


Figura 5.27: Detección de fuga en el Tanque 1 de la planta cuando existen fugas en los dos tanques utilizando el controlador FGS-PID para la Válvula 2

Figura 5.28: Detección de fuga en el Tanque 2 de la planta cuando existen fugas en los dos tanques utilizando el controlador FGS-PID para la Válvula 2

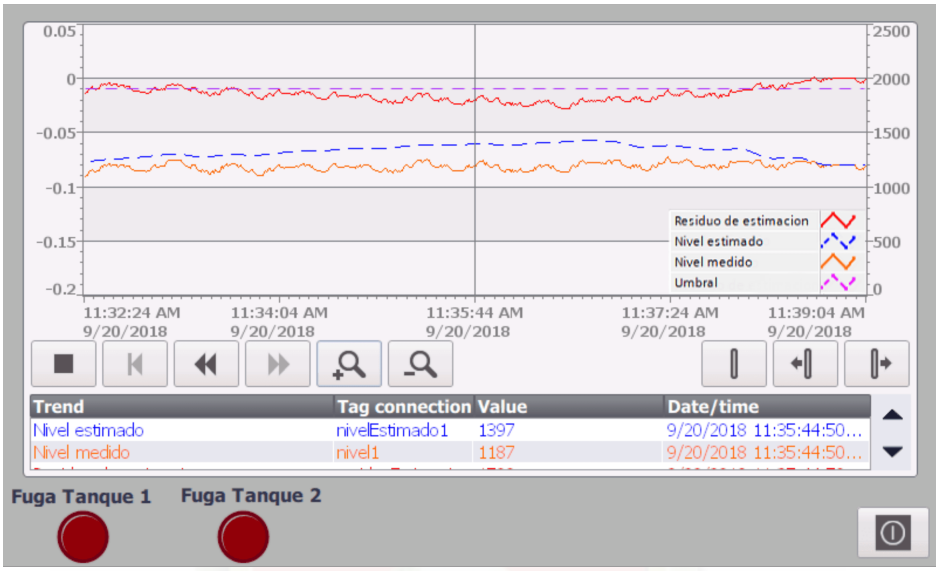


Figura 5.29: Recuperación de la estimación del Tanque 1 en la planta al quitar las fugas en los dos tanques utilizando el controlador FGS-PID para la Válvula 2

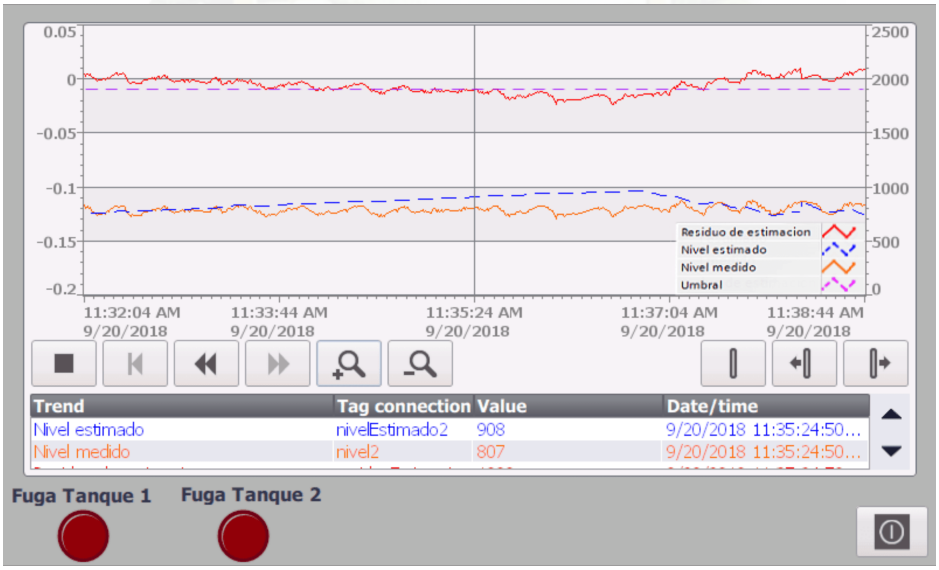


Figura 5.30: Recuperación de la estimación del Tanque 2 en la planta al quitar las fugas en los dos tanques utilizando el controlador FGS-PID para la Válvula 2

Tabla 5.2: Desempeño del sistema de detección de fallas

Esquema de control	Lugar de la fuga	Tiempo de detección de presencia de fugas (Activación de alarmas)	Tiempo de detección de recuperación de fugas (Desactivación de alarmas)
PID	Tanque 1	70.4 ± 13.33 seg	134.6 ± 44.19 seg
	Tanque 2	449.8 ± 106.54 seg	72.2 ± 7.63 seg
	Tanques 1 y 2	28.6 ± 7.14 seg *	94.6 ± 30.2 seg *
		164.6 ± 25.95 seg ‡	94.2 ± 21.81 seg ‡
FGS-PID	Tanque 1	105.8 ± 40.42 seg	65 ± 8.2 seg
	Tanque 2	133 ± 34.06 seg	35.8 ± 11.37 seg
	Tanques 1 y 2	79.2 ± 29.62 seg *	123 ± 45.23 seg *
		121.8 ± 26.59 seg ‡	53.4 ± 15.44 seg ‡

* Valor para el Tanque 1 cuando existen fugas simultáneas en los tanques 1 y 2.
‡ Valor para el Tanque 2 cuando existen fugas simultáneas en los tanques 1 y 2.

Evaluando el desempeño del sistema de detección de fallas, se puede notar que la activación de alarmas es más rápida en el Tanque 1 que en el Tanque 2. Al utilizar el esquema PID la activación de alarmas en el Tanque 1 es 84.34 % más rápida que en el Tanque 2, mientras que al utilizar el esquema FGS-PID la activación de alarmas en el Tanque 1 es 20.45 % más rápida. Esta diferencia considerable en los tiempos de detección de las fugas en cada tanque se debe a que el flujo de la fuga en el Tanque 1 es mayor al flujo de la fuga en el Tanque 2. Esto ocurre debido a que el flujo de la fuga es proporcional al nivel del tanque. Además el acoplamiento que presenta el Tanque 2 con respecto al Tanque 1 provoca que la estimación del EKF para el nivel del Tanque 2 tarde más tiempo en diferenciarse de las mediciones reales del nivel en el Tanque 2. Por su lado, el Tanque 1 es aislado del comportamiento del Tanque 2. Esto provoca que la detección de fugas en el Tanque 1 sea más rápida que en el Tanque 2. El aislamiento del Tanque 1 y el acoplamiento que presenta el Tanque 2 puede evidenciarse en las matrices de espacio de estados que se presentan en la Ecuación 4.15.

La Figura 5.31 presenta de manera visual los intervalos de confianza del tiempo de activación de alarmas. En dicha figura las etiquetas con * en el eje x indican que existen fugas simultáneas en los dos tanques. Se puede observar en el caso del Tanque 2, el uso del controlador FGS-PID reduce considerablemente el tiempo de activación de alarmas en un 51.33 %. Por otra parte, en el caso del Tanque 1, cuando existen fugas simultáneas el uso del controlador FGS-PID incrementa en un 27.93 % el tiempo de activación de alarmas. En los otros casos debido al solapamiento entre los intervalos de confianza se considera que los tiempos son iguales utilizando los dos

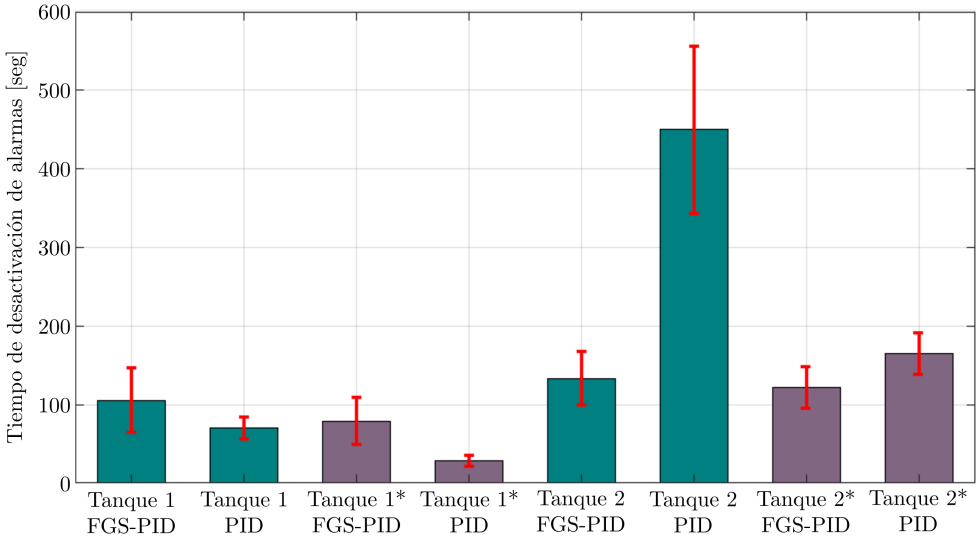


Figura 5.31: Intervalos de confianza en el tiempo de activación de alarmas de fuga

esquemas de control planteados.

En la Figura 5.32 se muestra de manera visual los intervalos de confianza del tiempo de desactivación de alarmas. Se puede observar en el caso que existen fugas simultáneas, el tiempo de desactivación de alarmas en el Tanque 1 es muy similar al utilizar los dos esquemas de control. En los otros casos se observa que el controlador FGS-PID reduce estos tiempos de desactivación de alarmas. En el caso del Tanque 1 se reduce en un 19.04 %. En el caso del Tanque 2 se reduce en un 26.96 %. En el caso del Tanque 2 cuando existen fugas simultáneas se reduce en 4.9 %.

En base a los intervalos de confianza de los tiempos de activación y desactivación de alarmas de fuga, se puede observar que estos tiempos son variantes en rangos amplios. Estas variaciones se deben a que el estado de las variables de la planta resulta crucial en el comportamiento del sistema de detección de fallos. Según los resultados presentados se puede concluir que el rendimiento de la activación y desactivación de las alarmas dependerá tanto del estado en que se encuentren las variables de la planta y el tipo de controlador utilizado.

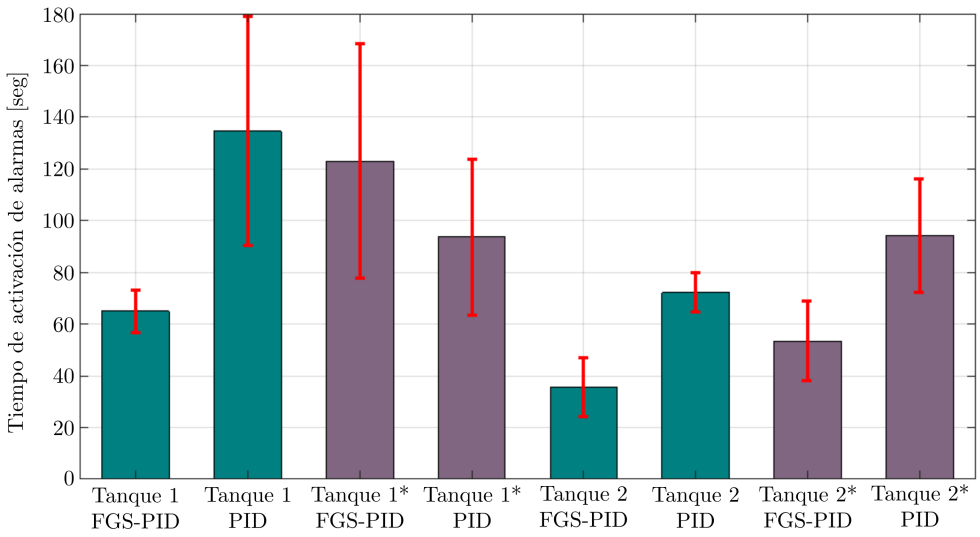


Figura 5.32: Intervalos de confianza en el tiempo de desactivación de alarmas de fuga

5.3. Comunicación con el servidor OPC-UA

El uso de *Modbus* para el manejo de las variables de proceso permite la interacción del sistema con el servidor OPC-UA desarrollado en [56]. Para integrar la información de la planta con el servidor OPC-UA, en primer lugar es necesario crear el dispositivo dentro del servidor. En la Figura 5.33 se muestra la ventana de configuración del *PiXtend* en el servidor. Al crear el dispositivo se indica que se utiliza *Modbus TCP* para la comunicación, se ingresa el nombre del dispositivo y su dirección IP. Adicionalmente se debe indicar el tamaño del bloque de registros de *Modbus*. Para el caso del sistema multi-tanque se emplean 18 variables ubicadas en los registros de retención. Finalmente, se define el tiempo de comunicación con el dispositivo. Este último se estableció en 500 ms debido a que es el periodo de muestreo del sistema. Por otra parte, se establece un margen de tiempo de 3 segundos para determinar posibles fallos en la comunicación con el dispositivo.

El siguiente paso consiste en crear las etiquetas (*tags*) dentro del servidor, los cuales representan las variables del proceso. En la Figura 5.34 se muestra la ventana de configuración de un *tag*. Para estas variables se requiere ingresar un nombre, su dirección dentro de los registros *Modbus*, el tipo de dato configurado como entero de 16 bits y determinar si el *tag* es de lectura y escritura o únicamente de lectura. En la Figura 5.35 se muestra en detalle un grupo de *tags* configurados en el servidor OPC-UA.

Para visualizar los valores de los *tags* en el servidor, estos deben ser suscritos a cambios. En la Figura 5.36 se muestra la ventana de visualización de *tags* suscritos en el servidor, se puede

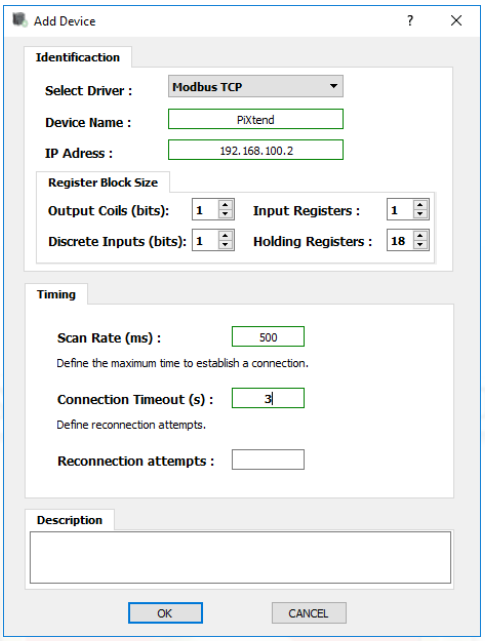


Figura 5.33: Creación del PLC *PiXtend* en el servidor OPC-UA

observar que es posible obtener los valores de las variables de proceso generadas por el *PiXtend*.

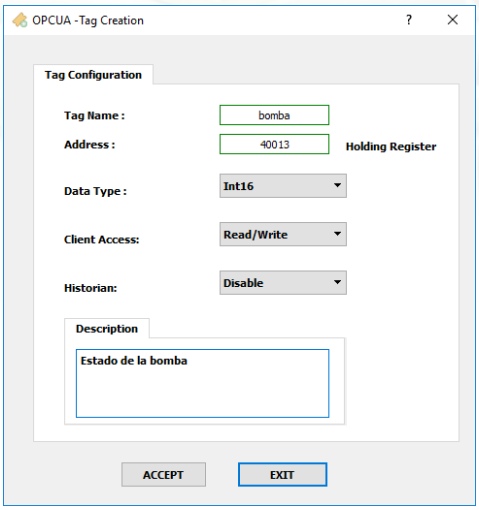


Figura 5.34: Creación de un *tag* en el servidor OPC-UA

5.4. Integración con un sistema SCADA de software libre

Aunque el alcance de este trabajo no incluye el desarrollo de un sistema SCADA, se ha evaluado una primera aproximación de la integración de la planta a un sistema SCADA. Para esto se utilizó un primer avance del sistema SCADA desarrollado en software libre por el equipo de trabajo de [56] y [11]. Este sistema SCADA funciona como un cliente OPC-UA y tiene las funcionalidades de poner en marcha o detener el proceso de la planta, así como modificar las referencias de los tanques y la visualización tanto de los niveles de los tanques como del flujo de entrada y salida del sistema. La adquisición de datos se realiza por medio de OPC-UA. Para esto, las variables del sistema SCADA se configuran en base a las direcciones de los nodos que representan dichas variables en el servidor OPC-UA. En la Figura 5.37 se muestra la ventana inicial del sistema SCADA en la cual se ingresa la dirección IP y puerto del servidor OPC-UA. En la Figura 5.38 se muestra la ventana principal del sistema SCADA. En la misma se puede observar los controles para iniciar o detener el sistema y para modificar las referencias, así como la visualización de los niveles y flujos por medio de animaciones. Los niveles de los tanques se encuentran en cm, mientras que los flujos son presentados en lt/min.

Tag Name	Adress	Data Type	Server Id	Subs Changes	Description
ref1	40001	Int16	ns=2;i=12	SI	Referencia del tanque 1
ref2	40002	Int16	ns=2;i=16	SI	Referencia del tanque 2
tipoCtrl	40003	Int16	ns=2;i=20	SI	Tipo de controlador para la valvula 2
nivel1	40004	Int16	ns=2;i=24	SI	Nivel medido del tanque 1
nivel2	40005	Int16	ns=2;i=28	SI	Nivel medido del tanque 2

Figura 5.35: *Tags* configurados en el servidor OPC-UA

Subscription Table

	SUBSCRIPTION DATE	TIME	DEVICE	TAG	TYPE	VALUE
1	2018/09/19	14:55:29:4155	PiXtend	ref1	Int16	12
2	2018/09/19	14:55:30:8745	PiXtend	ref2	Int16	8
3	2018/09/19	14:53:22:0861	PiXtend	tipoCtrl	Int16	0
4	2018/09/19	14:56:04:2938	PiXtend	nivel1	Int16	816
5	2018/09/19	14:56:04:3038	PiXtend	nivel2	Int16	424
6	2018/09/19	14:55:30:8825	PiXtend	sControl1	Int16	10000

Figura 5.36: Visualización de los valores de los *tags* suscritos a cambios

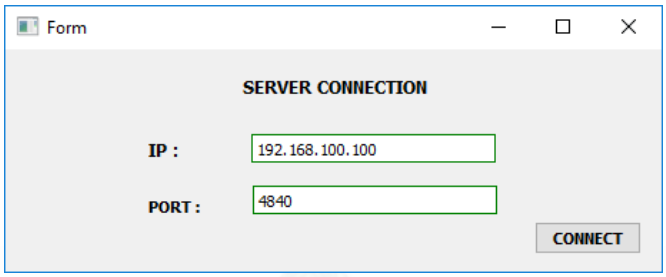


Figura 5.37: Ventana de inicio del sistema SCADA

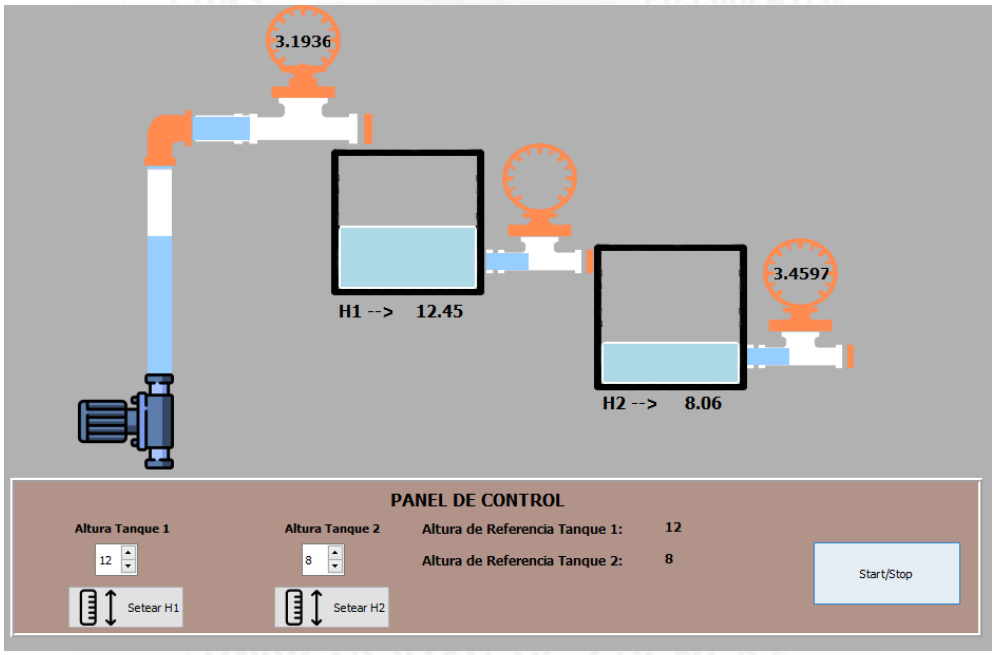


Figura 5.38: Ventana prinicpal del sistema SCADA

Capítulo 6

Conclusiones y recomendaciones

En este capítulo se presentan las conclusiones finales del trabajo realizado así como la interpretación de los resultados, las limitaciones que se encontraron en el proceso, y finalmente, se proponen maneras de ampliar la investigación a futuro.

6.1. Conclusiones y recomendaciones

Dentro del paradigma de la Industria 4.0, la interoperabilidad entre los distintos dispositivos que forman parte de los sectores de producción es un requisito imprescindible, una alternativa para cumplir este requerimiento es utilizar un servidor OPC-UA compatible con los dispositivos de control de la planta. De esta manera se puede asegurar la interoperabilidad y la escalabilidad con otros dispositivos, plantas o *software*.

Los dispositivos de bajo costo y hardware libre como el PLC *PiXtend* y el PLC *M-Duino* tienen sus limitaciones como velocidad de procesamiento, memoria, velocidad de lectura de los puertos de entrada como es el caso del PLC *PiXtend*. A pesar de las limitaciones que tienen estos PLCs de bajo costo, se ha demostrado en este trabajo que son dispositivos útiles para potenciar sistemas de control, comunicación y automatización de procesos de producción de pequeña y de mediana envergadura.

En una planta hidráulica es necesario instalar válvulas de seguridad *check* y válvulas de alivio de presión. Las válvulas *check* se ubican a la salida de la bomba y ayudan a proteger a la misma de golpes de ariete. Las válvulas de alivio de presión son necesarias en sistemas que poseen una válvula conectada con una bomba, ya que permiten regular la presión generada al cerrar una válvula en un sistema hidráulico, desfogando el agua que causa el incremento de presión generalmente hacia el tanque de bombeo. Por ello este tipo de válvulas se ubican entre la bomba y la primer válvula del sistema para permitir el desfogue de agua cuando existe un incremento de presión al cerrar la válvula.

El diseño del diagrama P&ID es muy importante para entender un proceso de producción, por tanto es la base para posteriores diseños e implementaciones de sistemas de automatización industrial de un proceso de producción.

El modelar matemáticamente una planta es imprescindible al momento del diseño de un sistema de control y automatización de un proceso de producción industrial. Esto permite simular el comportamiento del proceso y facilita el diseño de esquemas de control sin necesidad de intervenir directamente sobre la planta. Hay que tener en cuenta que un modelo matemático resulta una aproximación del comportamiento real de la planta, por tanto muchas veces resultará necesario hacer correcciones en los esquemas de control o una nueva sintonización de los parámetros que gobiernan el esquema de control.

En la prueba de funcionamiento de la planta real resultó necesaria la calibración de los instrumentos de medición y de las válvulas de regulación de caudal. Los módulos HC-SR04 resultan muy sensibles al movimiento ondulatorio que presenta el agua debido al caudal de ingreso en los tanques, por tanto es necesario contrarrestar este comportamiento. Por otro lado, también es

importante evitar objetos, como salpicaduras, que se interpongan entre el modulo HC-SR04 y el agua reposada a medir. Las válvulas de regulación de caudal también tienen que ser calibradas debido a que el porcentaje de apertura no es lineal. Además en la planta se presentan pérdidas debido a las conexiones de tuberías que afectan considerablemente el área eficaz de apertura de las válvulas.

En este trabajo se ha evaluado dos esquemas de control con detección de fallos: un esquema conformado por controladores PID para las tres válvulas que controlan los flujos de entrada y salida del sistema multi-tanque, y otro esquema similar en el que se implementa un FGS-PID que controla la apertura de la válvula que limita el flujo de salida del primer tanque de reserva. Los dos esquemas tienen un buen rendimiento en la planta real, sin embargo el esquema que utiliza el controlador FGS-PID permite reducir el sobreimpulso del Tanque 2 en un 45.71 % y el tiempo de estabilización un 12.76 %. Por otro lado se logró la detección y diagnóstico de fallas mediante la implementación de un EKF como un observador de estados. Para lograr la detección de fallos con el EKF se necesita encontrar una familia de modelos lineales que describan el comportamiento real del sistema. En este contexto se observa que el uso del controlador FGS-PID permite reducir el tiempo de activación de alarmas de fuga en el Tanque 2 en un 51.33 % con respecto al uso del controlador PID en la Válvula 2, aunque el tiempo de activación de alarma en el Tanque 1 se incrementa en 33.45 %. Por otra parte, el uso del controlador FGS-PID permite reducir el tiempo de desactivación de alarmas un 19.04 % en el Tanque 1 y un 26.96 % en el Tanque 2. Por otra parte, según los intervalos de confianza, no siempre estos resultados son característicos. Por tanto, el rendimiento de la activación y desactivación de las alarmas dependerá tanto del estado en que se encuentren las variables de la planta y el tipo de controlador utilizado.

6.2. Trabajo futuro

- Integrar la planta desarrollada a un sistema SCADA complejo de software libre compatible con el servidor OPC-UA.
- Evaluar otros esquemas de control y sistemas de detección de fallos en la planta.
- Realizar una mayor cantidad de pruebas en los esquemas de control y sistemas de detección de fallos con el fin de reducir los intervalos de confianza.
- Evaluar la escalabilidad del sistema de comunicación integrando equipos de distintos fabricantes.
- Integrar al servidor OPC-UA distintos protocolos de comunicación industrial como CAN.
- Evaluar nuevas tendencias de redes definidas por software y servicios en la nube para su aplicación en sistemas industriales.

Apéndices

FONS VITÆ ERUDITIO POSSIDENTIS
UNIVERSIDAD DE CUENCA

Apéndice A

Diseño de los tanques y armazón de la planta del sistema multi-tanque

En este apéndice se presentan las láminas técnicas del diseño de la planta del sistema multi-tanque. En este apéndice se encuentra la lámina del montaje completo de la planta (incluyendo actuadores, sensores y tuberías), así como, láminas técnicas del armazón de la planta, del tanque de bombeo y los tanque de reserva.

A6
105x148

A5
148x210

A4
210x297

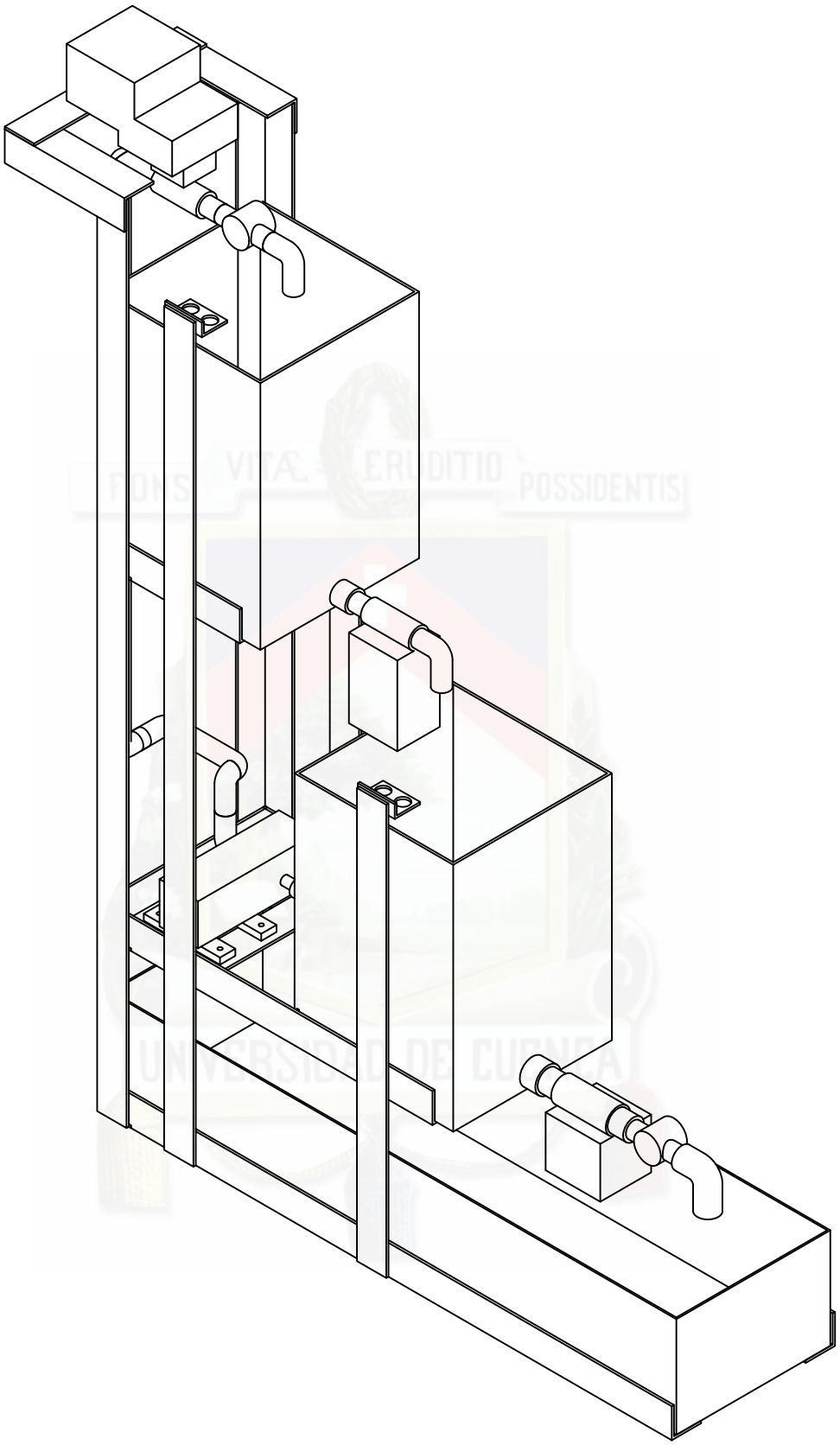
A3
297x420

A2
420x594

A1
594x841

A0
841x1189

Hoja de enseñanza Técnica



	Nombre	Fecha	Escuela de Ingeniería Electrónica y Telecomunicaciones	UNIVERSIDAD DE CUENCA
Dibujado por:	Ricardo Enderica Fabricio López	25/06/2018		
Proyección	PLANTA DEL SISTEMA MULTITANQUE			Escala: 1:6 Lámina N° 1

A6
105x148

A5
148x210

A4
210x297

A3
297x420

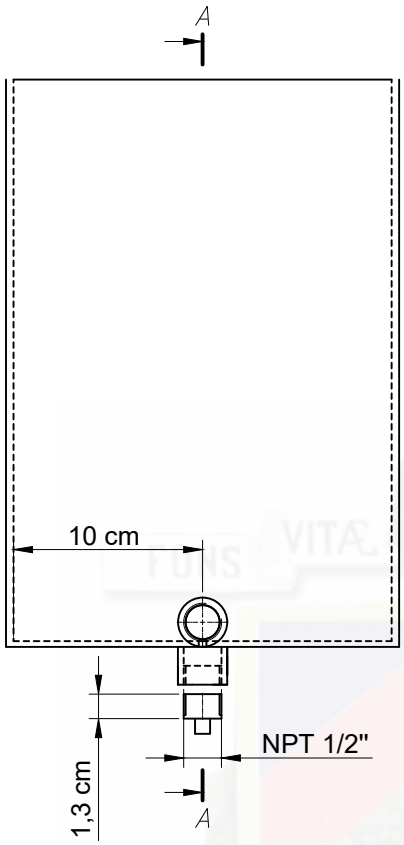
A2
420x594

A1
594x841

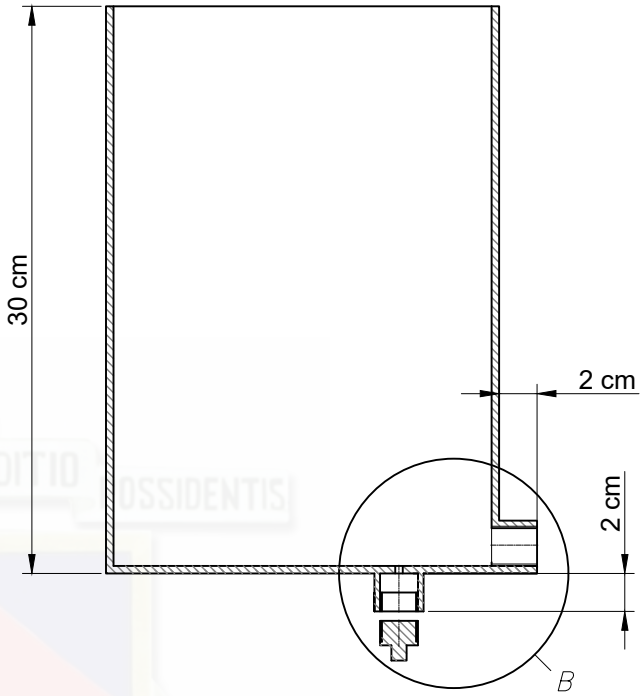
A0
841x1189

Hoja de enseñanza Técnica

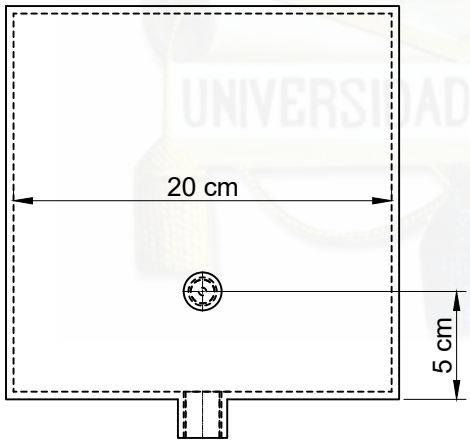
Vista frontal



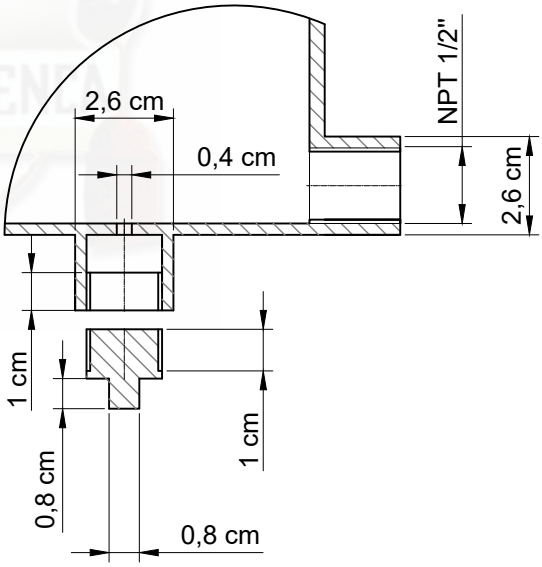
Corte A–A



Vista inferior



B (1:2)



	Nombre	Fecha	Escuela de Ingeniería Electrónica y Telecomunicaciones	UNIVERSIDAD DE CUENCA
Dibujado por:	Ricardo Enderica Fabricio López	25/06/2018		
Proyección	TANQUE DE RESERVA 1		Escala: 1:4	
			Lámina N° 2	

A6
105x148

A5
148x210

A4
210x297

A3
297x420

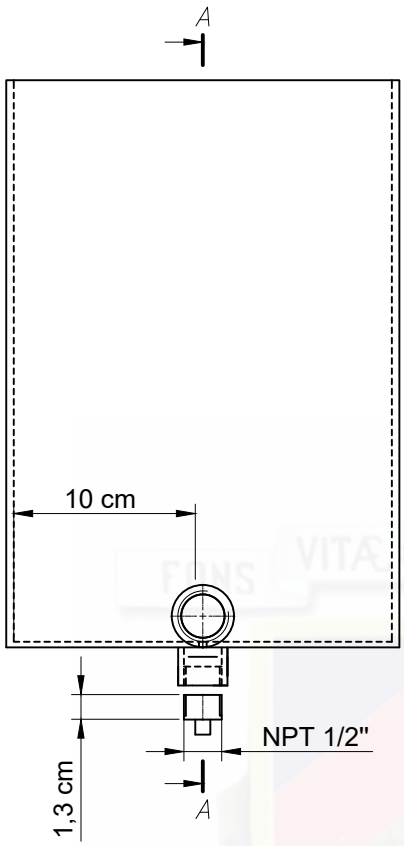
A2
420x594

A1
594x841

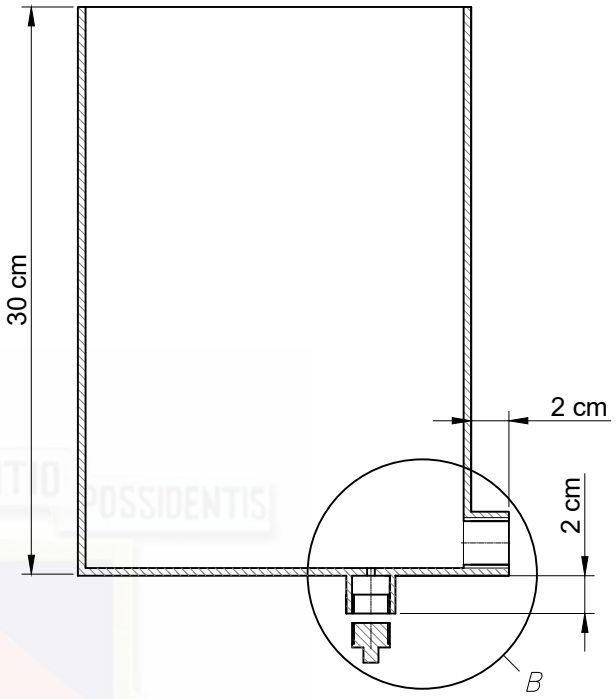
A0
841x1189

Hoja de enseñanza Técnica

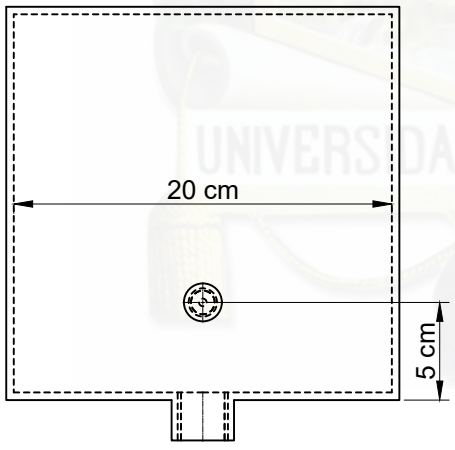
Vista frontal



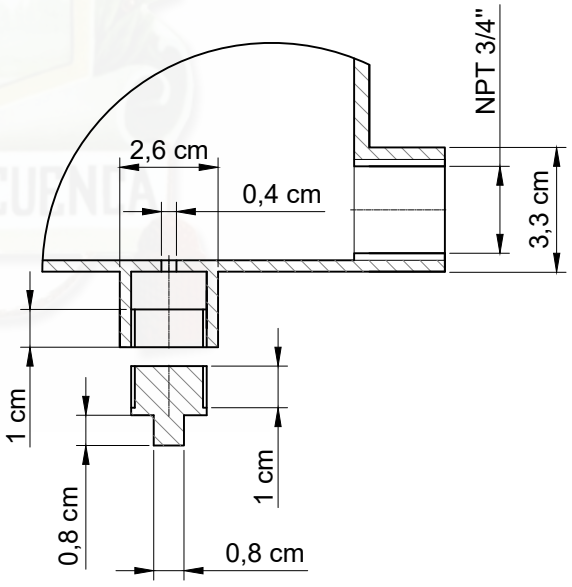
Corte A–A



Vista inferior



B (1:2)



	Nombre	Fecha	Escuela de Ingeniería Electrónica y Telecomunicaciones	UNIVERSIDAD DE CUENCA
Dibujado por:	Ricardo Enderica Fabricio López	25/06/2018		
Proyección	TANQUE DE RESERVA 2		Escala: 1:4	
			Lámina N° 3	

A6
105x148

A5
148x210

A4
210x297

A3
297x420

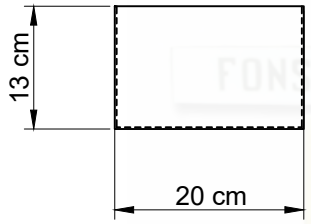
A2
420x594

A1
594x841

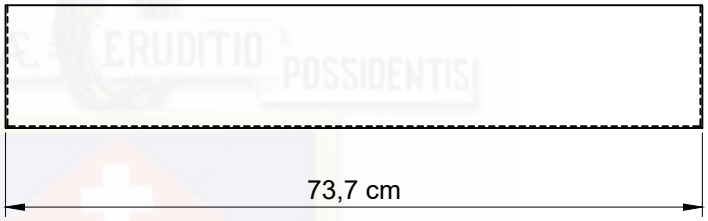
A0
841x1189

Hoja de enseñanza Técnica

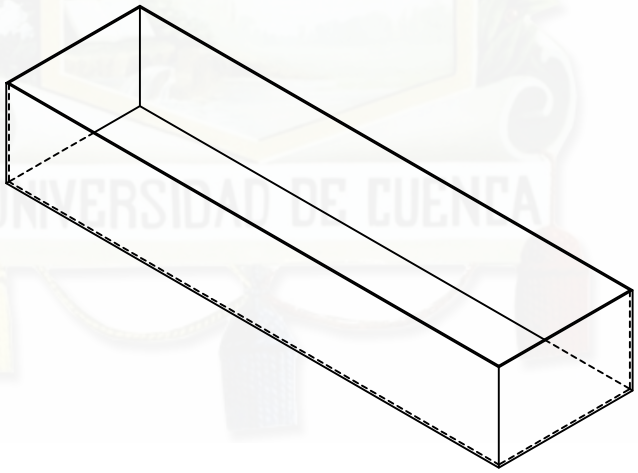
Vista frontal



Vista lateral



Vista isométrica



	Nombre	Fecha	Escuela de Ingeniería Electrónica y Telecomunicaciones	UNIVERSIDAD DE CUENCA
Dibujado por:	Ricardo Enderica Fabricio López	25/06/2018		
Proyección	TANQUE DE BOMBEO		Escala: 1:8	
			Lámina N° 4	

A6
105x148

A5
148x210

A4
210x297

A3
297x420

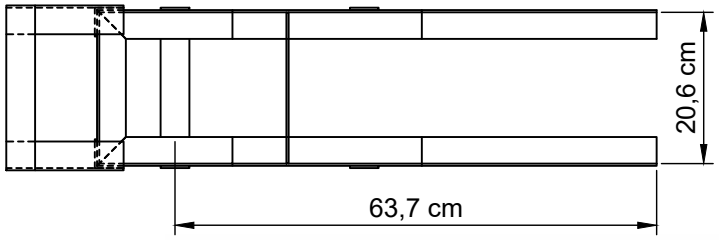
A2
420x594

A1
594x841

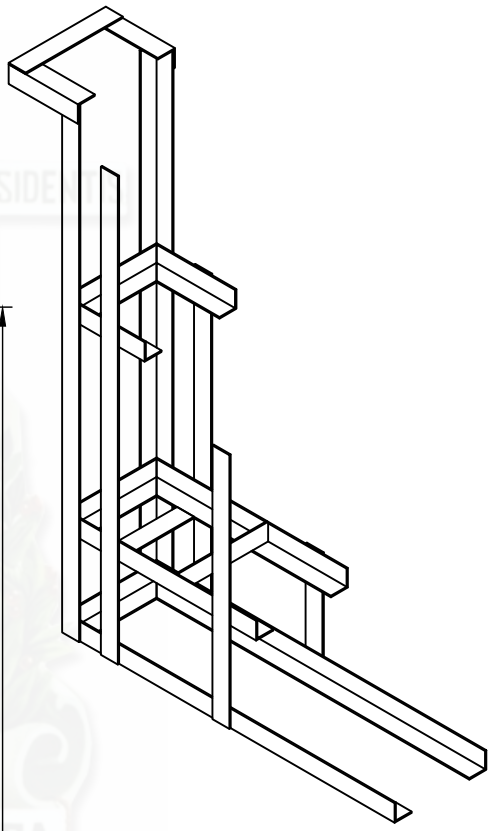
A0
841x1189

Hoja de enseñanza Técnica

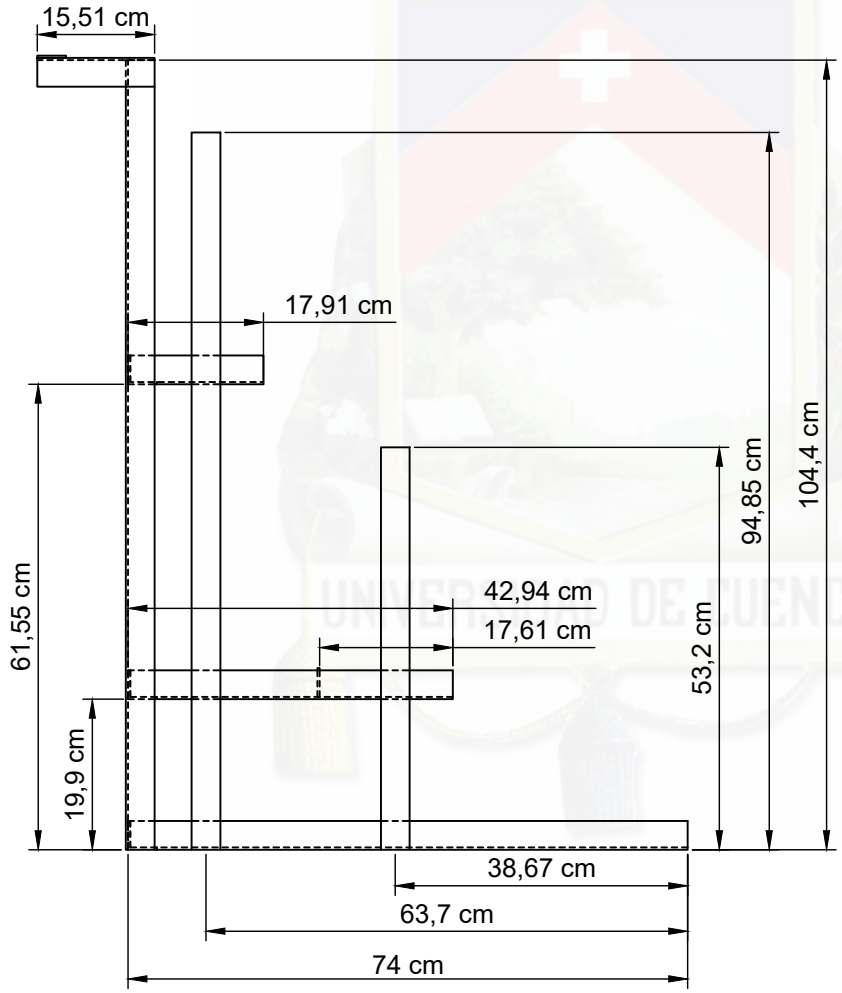
Vista superior



Vista isométrica
Escala: 1:12



Vista lateral



	Nombre	Fecha	Escuela de Ingeniería Electrónica y Telecomunicaciones	UNIVERSIDAD DE CUENCA
Dibujado por:	Ricardo Enderica Fabricio López	25/06/2018		
Proyección	ARMAZÓN DE LA PLANTA			Escala: 1:10 Lámina N° 5

Apéndice B

Diseño del tablero de instrumentación para sistema multi-tanque

En este apéndice se presentan las laminas técnicas del diseño del tablero de instrumentación. En este apéndice se encuentran las láminas del montaje completo del tablero (incluyendo fuentes de alimentación, PLCs, canaletas y rieles), así como, láminas técnicas con las dimensiones del tablero.

A6
105x148

A5
148x210

A4
210x297

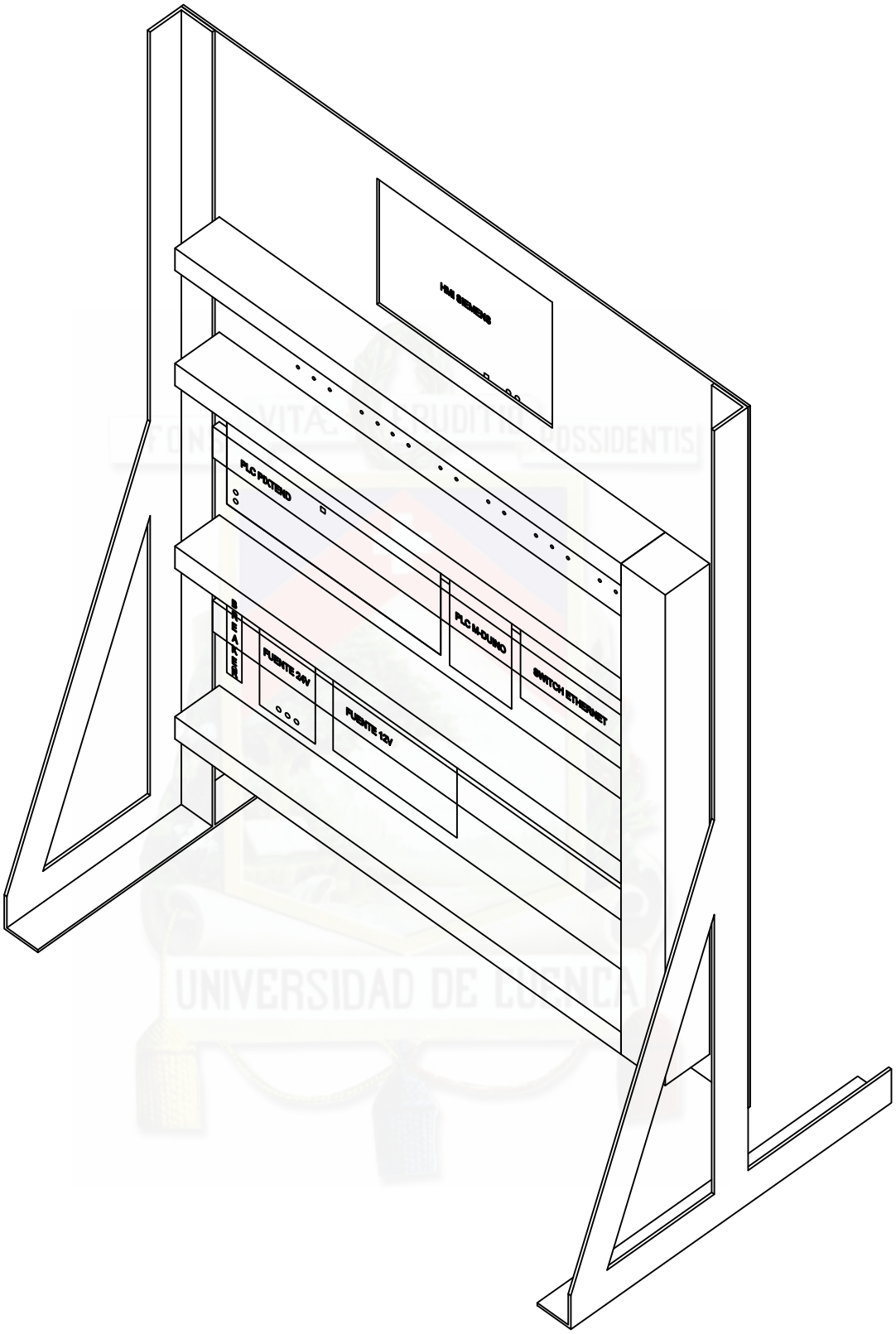
A3
297x420

A2
420x594

A1
594x841

A0
841x1189

Hoja de enseñanza Técnica



	Nombre	Fecha	Escuela de Ingeniería Electrónica y Telecomunicaciones	UNIVERSIDAD DE CUENCA
Dibujado por:	Ricardo Enderica Fabricio López	25/06/2018		
Proyección	MONTAJE DEL TABLERO DE INSTRUMENTACIÓN		Escala: 1:5	
			Lámina N° 1	

A6
105x148

A5
148x210

A4
210x297

A3
297x420

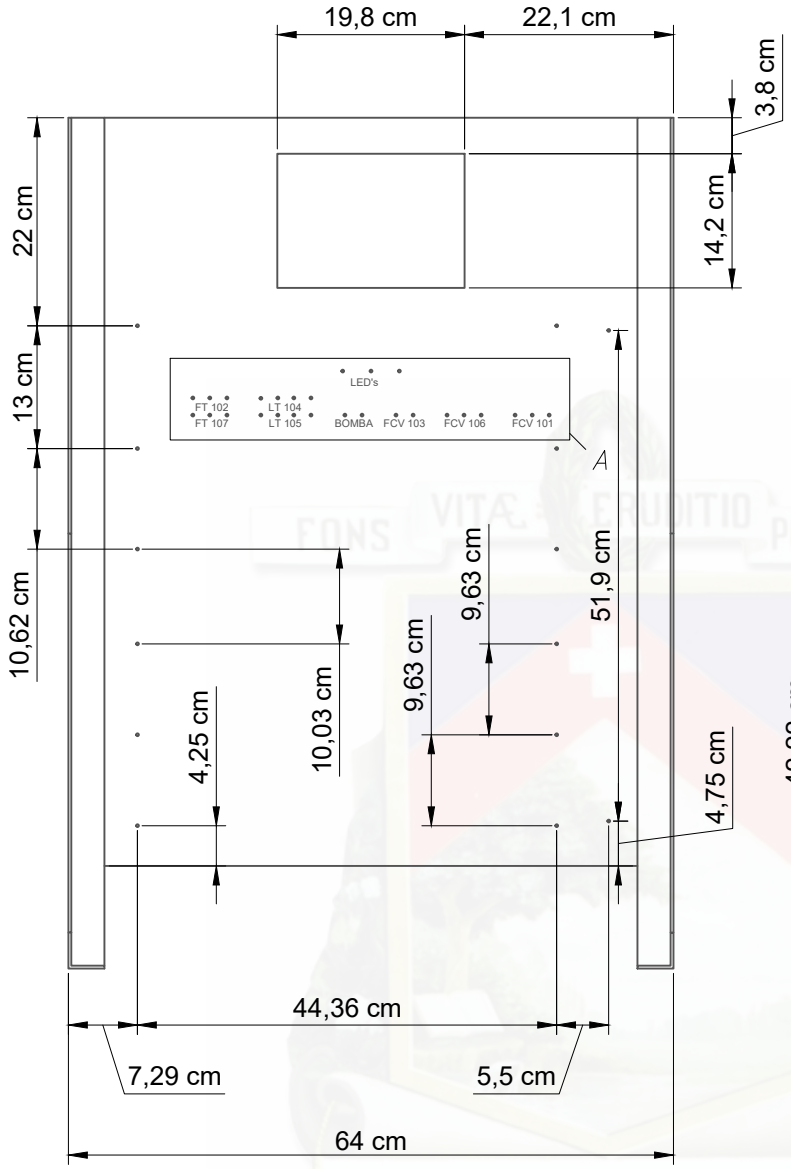
A2
420x594

A1
594x841

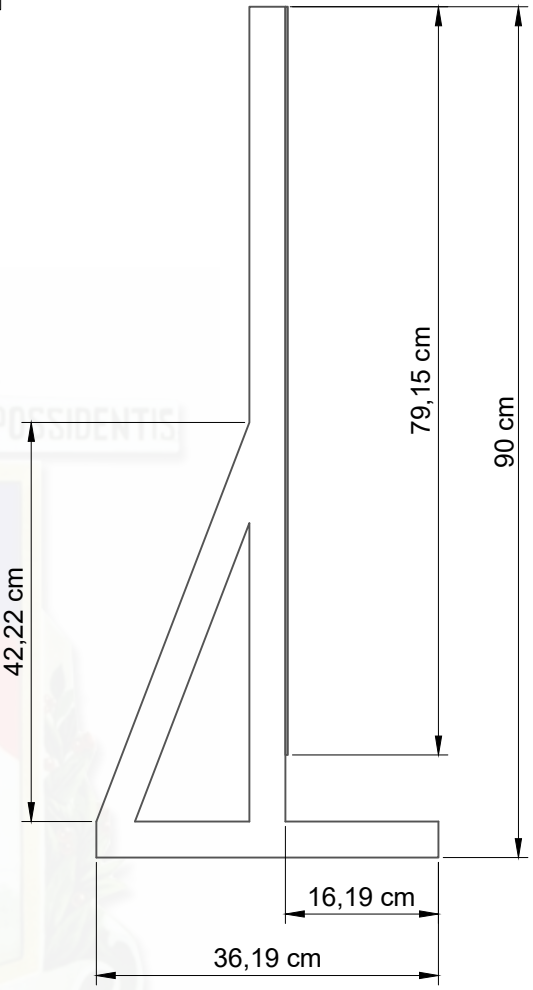
A0
841x1189

Hoja de enseñanza Técnica

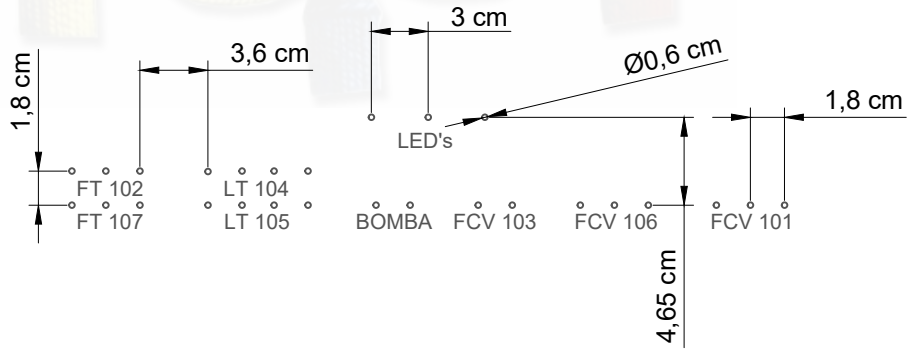
Vista posterior



Vista lateral



A (1:4)



	Nombre	Fecha		
Dibujado por:	Ricardo Enderica Fabricio López	25/06/2018	Escuela de Ingeniería Electrónica y Telecomunicaciones	UNIVERSIDAD DE CUENCA
Proyección	TABLERO DE INSTRUMENTACIÓN		Escala: 1:8	Lámina N° 2

Apéndice C

Manual de usuario del Pixtend y la librería de programación de Pixtend para Python

En este apéndice se presentan las especificaciones técnicas del PLC PiXtend V1.3 y de la librería para la programación de PiXtend mediante Python.

PiXtend V1.3 Data sheet

Technical Data and connection hints

Stand 15.06.2016, V1.05

Qube Solutions UG (limited liability)
Arbachtalstr. 6, 72800 Eningen, Germany
http://www.qube-solutions.de
http://www.pixtend.de

Technical Data – PiXtend- power supply (internal):

Characteristic	Value	Comment
Typ	Switching regulator	Step-down / buck converter
Switching frequency	52 kHz	Fixed frequency, internal oscillator
Input voltage	12-24 V DC	Rated input voltage
	30 V DC	Maximal input voltage
Output voltage	5 V DC	+/- 5 % of the rated voltage
Output current	max. 2 A	
Output ripple	typ. < 10 mV	
Energy reserves	min. 10 ms	at 24 V supply and 2 A output current
Short circuit- and overload protection	yes	thermal at 3 A, self-resetting
cooling	yes	passive cooling element
Reverse polarity protection	yes	Up to -30 V
Potential separation	no	
Status indication	LED (green)	Labeling: "+5V"
Maximum cable length	< 3 m	

Warning!

Operate the PiXtend only with the defined voltage- and load ranges. A constant overload can cause lasting damages to the electrical components.

Danger of burns!

According to load and ambient temperature, the voltage regulator, cooling element and diodes of the power supply can have temperatures up to 75 °C

Direct touching should be avoided!

3. PiXtend- Microcontroller

Figure 7: PiXtend microcontroller

The PiXtend- Controller is an 8 bit- RISC- Processor, the ATmega32A from the Atmel Corporation. The Atmega- series is very popular and widespread. Similar controller you can find for example on the Arduino-boards or on our LED-Qube 5.

The microcontroller takes over multiple tasks

- Control the digital outputs and relays
- Generates servo- and PWM- signals
- Reading of analogue and digital inputs
- Operation of the 4 GPIOs (as input, output or temperature- and humidity sensors (DHT11/22 / AM2302))
- Signal- and data processing
- Watchdog-functionality and voltage monitoring (drown out- detection), data security layer with 16 bit CRC- checksum

Raspberry Pi and PiXtend are connected over the SPI- bus (serial peripheral interface). The Raspberry Pi is the bus-master, the PiXtend- controller is the slave.

Our open source firmware is written in the programming language „C“ and compiled with the free of charge program AVC-GCC- Compiler. If needed the controller can be reprogrammed as wanted. But please consider that only with the original firmware the function and the CE- conformity can be guaranteed.

Warning!

When connecting an external switch or push button the greatest caution is necessary!

It only should be connected and pressed a switching element at the GPIO23 (Pin 16), if the GPIO is configured as input. Otherwise the Raspberry Pi can be damaged.

Furthermore, as usual at the Raspberry Pi, it is not allowed to apply higher voltages as 3.3V to the GPIO23.

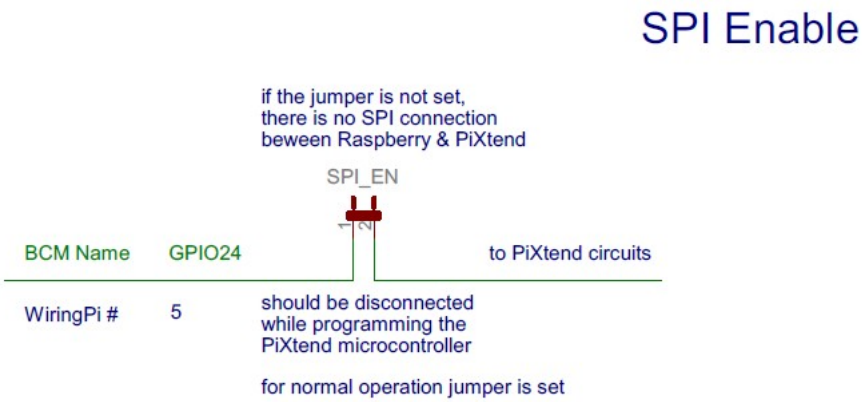


Figure 11: Principle-circuit diagram: SPI- Enable

The Jumper "SPI_EN" should be plugged always in normal operation. Or the data connection between Raspberry Pi and PiXtend is not possible.

But only the plugging of the jumper doesn't activate the data connection automatically. Additionally it is necessary to configure the GPIO24 (Pin 18) as output and set it to HIGH-level (3.3V)

In the PiXtend- test programs under Linux or with CODESYS the GPIO24 is set automatically to activate the data transfer between the controllers.

If the complete software is getting written by oneself (for advanced users) so the handling of the SPI- Enable should not be forgotten!

Technical Data - Jumper-position "24V":

Characteristic	Value	Comment
Type of inputs	Digital input, ohmic	For DC voltage
	Type 1 & 3 according to IEC 61131-2	
	positive switching	
	1-wire connection technology	with GND- reference
Rated voltage	24 V	
Voltage for high-level	min. 7 V	logic "1", HIGH-level
Voltage for low-level	max. 5 V	logic "0", LOW-level
Hysteresis	min. 1 V	
Input current at rated voltage	7.6 mA	
Maximal voltage	30 V	Input current: 9.7 mA
Polarity reversal protection	yes	Up to -30 V
Potential separation	no	
Status monitoring	LED (green)	
Maximum cable length	< 30 m	

Warning!

Voltages higher 30V DC can lead to overheating and the defect of of components.

The inputs are designed exclusively for DC voltages. It is not allowed to connect AC voltages.

Technical Data:

Characteristic	Value	Comment
Type of outputs	digital output, semiconductor	For DC voltage
	Each one open contact	SPST
	negative switching	N-channel power-mosfet
Rated current	3 A	
Short circuit current	max. 6 A	
Maximum voltage	30 V	
Maximum switching capacity	15 W 36 W 72 W	at 5 V DC at 12 V DC at 24 V DC
"ON"-resistor	about 100 mΩ	At logic "1"
Short circuit protection overload protection	yes yes	Self-resetting fuse (Polyfuse), thermal
Potential separation	no	
Status monitoring	LED (green)	
Maximum cable length	< 30 m	

Warning!

Voltages higher 30 V DC can lead to overheating and the defect of of components.

The outputs are designed exclusively for DC voltages. It is not allowed to connect AC voltages.

Danger of burns!

According to load and ambient temperature, the power transistors and fuses can have temperatures up to 75 °C.

Direct touching should be avoided! Output currents higher then 3 A are not allowed in standard operation.

Connection instructions

The following circuit diagram explains the connection of different loads and the digital outputs. On the left side (INT) is the internal circuit of the outputs and on the right side (EXT) is shown the possible external circuit.

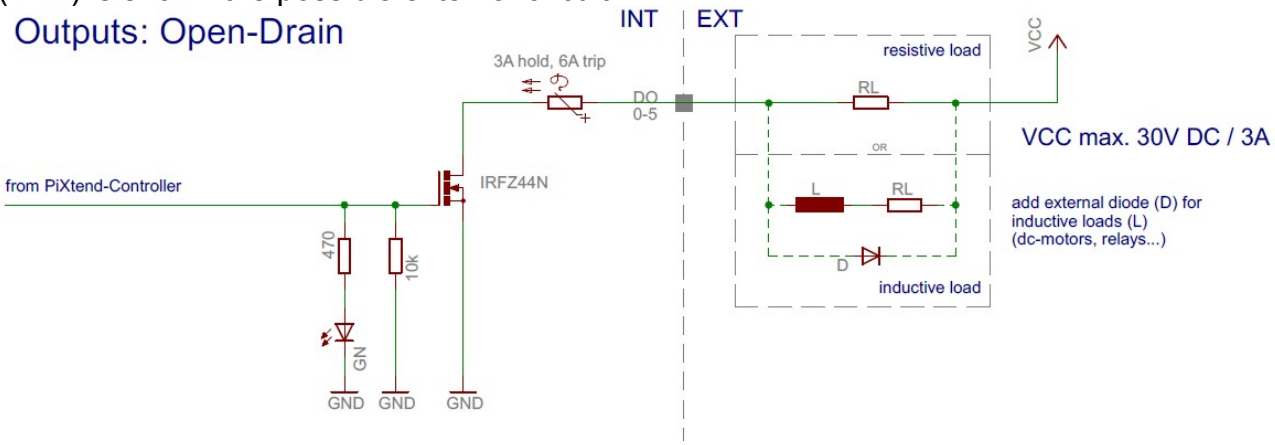


Figure 18: Principle-circuit diagram: Connection of the digital outputs

At inductive loads it needs a protection in the external circuit, to ensure that the voltage at the output never rises higher then 40 V. This can be realised for example with a free-wheel diode (1N4004), as it is shown in Figure 18.

Inductive loads are DC-motors, relays, contactors, solenoids etc.

Should the outputs get connected with inputs (at positive switching) of another controlling device, it will be needed an external pull-up resistor for the input voltage of the controlling device.

The ground connections (GND) of external power supply units are to connect directly with the GND- connections of the connector block of the digital outputs.

5.2 Analogue outputs

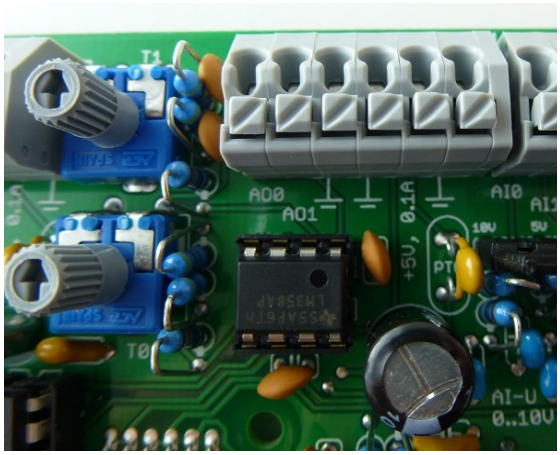


Figure 31: connection block - analog outputs

The full placement of PiXtend (assembly kit full / ARTC / ePLC) has two analogue voltage outputs. It is possible to output voltages in two channels, independent from each other, in areas from 0 to 10 V in 10 mV-steps.

Example of use

- Supply and controlling of small loads: Mini-DC-motor, LED(s)
- Function generator (Output of sinus, rectangle and triangle voltage, etc.)
- Connection with analogue inputs of other controlling devices and power amplifiers
- Controlling analogue monitoring device

Both outputs are short circuit protected and are able to run a current of 10 mA in normal operation. The analogue outputs are supplied over the central voltage supply of the PiXtend.

The fine trimming of the output voltage area is made by the two potentiometers T0 and T1 on the PiXtend.

Technical Data:

Characteristic	Value	Comment
Type of outputs	Analogue output after IEC 61131-2	DC voltage in positive area
Operating mode	Voltage output (AO-U)	
Conversion method	String DAC	One converter per channel
Rated voltage range	0..10 V DC	Can be trimmed to the range 0..5 V DC per potentiometer
Load impedance range	>= 1 kΩ	
Rated output current	10 mA	at 1 kΩ load, 10 V min. 13.5 V VCC
Digital resolution	10 bit	
Smallest digital step (LSB)	9.77 mV	Depending to the potentiometer setting
Maximum measuring failure at 25°C	+/- 2 % (+/- 0.2V)	of the value range, depending to the potentiometer setting
Temperature coefficient	+/- 0.025 % pro °C	at area 0 – 40 °C ambient temperature
Data format	16 bit Integer 32 bit Float	raw value (right-justified) output value in V
Short circuit protection	yes	Short circuit to GND
Maximum cable length	< 3 m < 30 m	unshielded cable shielded cable

Information!

Because the analogue outputs are supplied from the central PiXtend-power supply (12-24 V), following is to mind: The rated output current of 10 mA per channel can only be taken, if the PiXtend is supplied with min. 13.5 V.

6. Special in- and outputs

Some in- and outputs of PiXtend are having special functions.
The functions and the possibilities that they open up are described in the following.

6.1 PWM/Servo-outputs

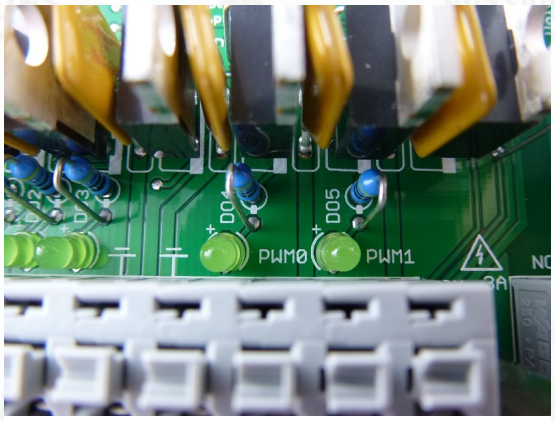


Figure 33: PWM/Servo-outputs

On the 10-pole connection block of the digital outputs are too two connectors named PWM0 and PWM1. At this special outputs it is possible to output pulse width modulated (PWM) signals with adjustable frequency and duty cycle or to directly connect a model making servo.

With help of the jumper „Dox- PWMx“ the PWM- signal directly layed at the digital outputs DO4 and DO5, so that it is possible to run loads up to 3 A.

Example of use

- control position of up to two model making servos
- rotation speed control of fans and other DC-motors
- dimming of DC-lamps and LEDs (high-power-LEDs too)
- controlling of heating element temperature
- fine adjustable clock source for a variety of electronical applications

Connection instructions

The following circuit diagram shows how to connect consumers and devices to the PWM or digital-outputs of PiXtend. On the left side (INT) is the internal circuit of the outputs, on the right side (EXT) is shown a possible external circuit.

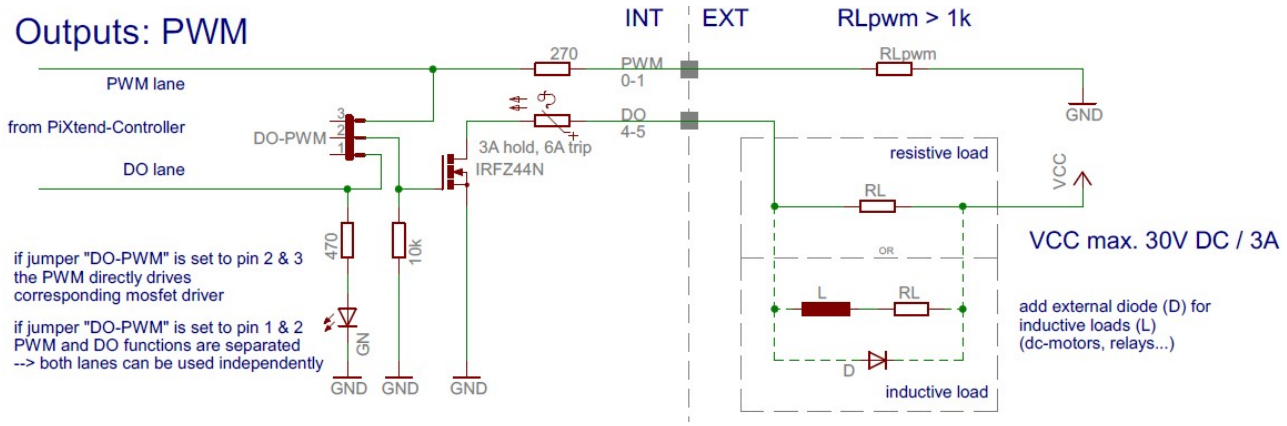


Figure 34: Principle-circuit diagram: Connection of the PWM- / Servo outputs

The PWM outputs PWM0 and PWM1 can be operated in 2 ways. The Jumper „Dox-PWMx“ is for the selection of the operation mode:

1. ***Jumper left (Pin 1 and 2 connected) → DO-mode***

   DO and PWM work independent from each other. At the PWM outputs can the PWM- signal be taken as 5 V-TTL-level. It is only allowed to charge the load with a resistor higher then 1 kΩ, connected with the high omic inputs of a servo motor or other another digital input.

2. ***Jumper right (Pin 2 and 3 connected) → PWM- mode***

   The PWM- signal is getting redirected to the power switch of the belonging output and is able to switch /tact the connected load of this DO because of that.
   The „normal“ DO-function is deactivated in this mode. The changing of the values from DO4 and DO5 in the software is not affecting the state of the respective power driver.

In the software (PiXtend-Linux- Tools and CODESYS- Demoprojekt) can be chosen between Servo- and PWM- mode:

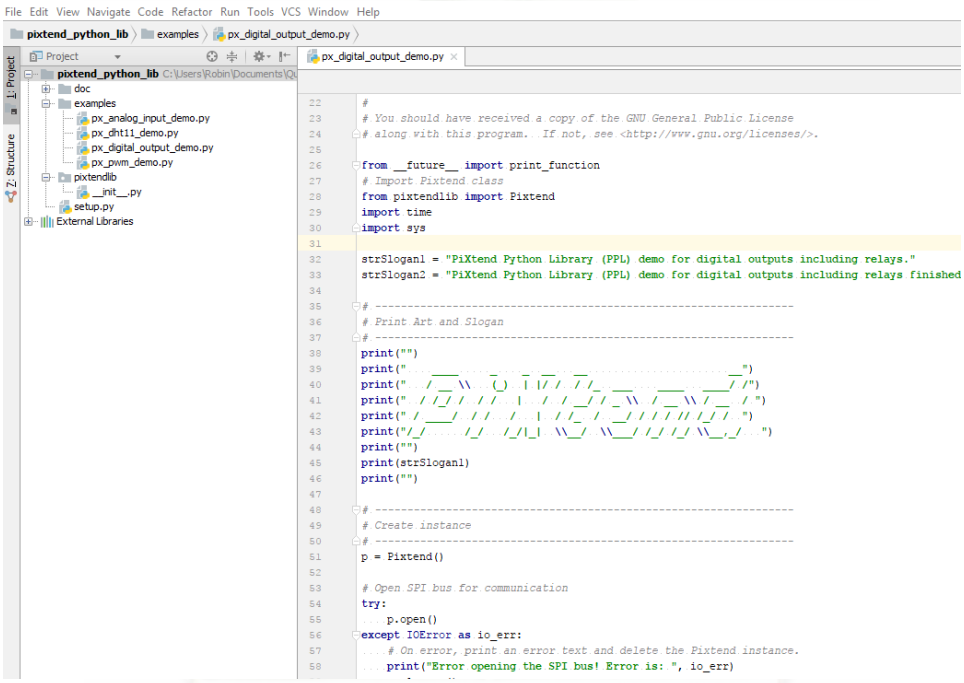
– Servo-Mode (standard-setting): Frequency 50 Hz (20 ms), 1-2 ms „ON“-time
  1ms: zero- position
  2ms: full deflection
– In PWM- Mode it is possible to configure Frequency and duty-cycle completely free.

It is only allowed to connect cables shorter then 3 meter to the connectors PWM0 and PWM1.

All further information to the controlling of the PWM- outputs by software are to look up in the according manuals / App-Notes. On the PiXtend- homepage (www.pixtend.de) are all documents ready to download for free.

Application-Note
PiXtend Python Library

Installation, Orientation and Programming



APP-PX-401

Status: 05.03.2018, V1.01

Qube Solutions UG (haftungsbeschränkt)
Arbachtalstr. 6, 72800 Eningen, Germany
http://www.qube-solutions.de/
https://www.pixtend.de

1.1 Requirements

The driver supports PiXtend with Python 2.7.9 and later versions (not Python 3) as well as **PiXtend V1.2 und V1.3**. You can use any PiXtend system.

Moreover, you are not set on for a particular model of Raspberry Pi. However, we recommend one of the following models: **B, B+, 2 B, 3 B**.

Download the ***PiXtend Image - PiXtend Python Library*** SD card image from our download section and use it as the initial point for your projects. Alternatively, you can also use the original Raspbian Jessie image. You will find the appropriate installation steps in chapter 3.

You can either access the Raspberry Pi by cable-connected keyboard and monitor, or by SSH (TeraTerm / Putty) from a computer. If you are using an original Raspbian Image directly, the Raspberry Pi needs an active Internet connection to carry out the steps shown in this document.

We further recommend keeping the app notes *Control- and Status-Bytes* ready as well as the *Process Data of PiXtend* ready. These documents contain information on the configuration of the PWM-outputs and analogue inputs as well as interesting facts about the GPIOs, the digital in- and outputs, and the PiXtend relays.

1.2 Disclaimer

Qube Solutions UG cannot be held responsible for any damages that may result from the use of the provided software, hardware, drivers or the steps described in this application note or software by third-party manufacturers.

1.3 Safety instructions

PiXtend must not be used in safety-critical systems.

Before use, please check the suitability of Raspberry Pi and PiXtend for your application.

3.2 PiXtend Python Library

Since all requirements for the installation of the *PPL* are fulfilled, in the first step we create a directory for the *PPL* package, then download the package and unpack it into the created directory. In the last step we install the *PPL* package. Once this is done, the *PiXtend Python Library* can be used globally in all Python 2.7.9 programs.

The installation of a Python package can produce a lot of output on the console, this is completely normal.

Execute the followings steps in sequence:

mkdir ppl

wget http://www.pixtend.de/files/downloads/ppl_v0.1.1.zip

unzip ppl_v0.1.1.zip -d ./ppl/

cd ppl

sudo python setup.py install

If the last command was successfully executed, the *PiXtend Python Library* is now installed and can be used in your own *Python* programs.

The installation of the PPL package can be verified with *pip freeze* as shown in the previous chapter. The entry pixtendlib == 0.1.1 can now be found in the list of installed packages.

All we need now is an editor to write *Python* programs, one to transfer the created programs to the Raspberry Pi and execute them.

NOTE:

*The functions and features of the PPL must not be called faster than every **100 ms**!*

4.1 The PiXtend Python Library (PPL)

So far, we have used the PiXtend Python Library together with an example. In order to create a program, some knowledge about the PPL itself is necessary, namely what properties are there and which functions must be used so that something is working at all.

The following is a brief description of the most important functions and properties that are required to work with the digital inputs and outputs. A complete overview of all properties and functions can be found in *Appendix A* of this document. We also recommend using the *WinSCP* program to download the *Python* examples from the Raspberry Pi and to look at them, e.g. if you would like to use the analog inputs and outputs, GPIOs or model-building servomotors. In the *PPL* folder on the Raspberry Pi is a folder named *doc, h*ere you will find a HTML file created with the *Python* program *pydoc*. This is a purely technical (automatically generated) documentation of the entire *PPL package*, which shows all public functions and properties along with comments from the source code.

The *PPL* essentially contains one Python class: the ***'Pixtend'*** class derived from a *Python* object. This class offers the user a set of properties and functions to configure the micro-controller on the *PiXtend* board and to work with the digital and analog inputs and outputs as well as the GPIOs.

The following table shows the most important functions and properties:[5]

Name	Type	Description
open	function	The *open* function starts the *Python* SPI driver, opens the SPI master 0 with chip select line 0, and sets the transfer speed to100 kHz required for the micro-controller. The *open* function must be called right after the creation of the PiXtend object, if another function or property is used, an IOError will be thrown. Example: p = Pixtend() p.open()
auto_mode	function	The *auto_mode* function facilitates the communication with the micro-controller and should be called next after the *open* function, preferably in an infinite loop or until the function returns 0 and the *uc_status* property has the value 1. From this point on, the *auto_mode* function can also be called sporadically, e.g. whenever new values are required or an output needs to be set. A cyclic call is recommended but is not mandatory. *There is one restriction: The auto_mode function must not be called faster than every 100 ms.* Example: if p.auto_mode() == 0: p.relay0 = p.ON
close	function	If the *Python* program is to be closed, it is recommended to call the *close* function first and then delete the PiXtend object subsequently. The *close* function resets all internal variables and objects and closes the SPI driver. This approach is to help to prevent memory leaks and to end the Python program *"cleanly"*. This way you can restart your Python program immediately without an error message.

[5] The meaning of (r) and (rw) behind a property: = read only and rw = read and write

		Example: p.close() p = None
digital_input0 .. 7	Property (r)	With these 8 *read only* properties (*digital_input0* to *digital_input7*), the states of the digital inputs can be read from the *PiXtend* board. The properties provide either the value 0 for OFF or the value 1 for ON.
digital_output0 .. 5	Property (rw)	The 6 digital outputs can be read as well as written to via the properties *digital_output0* to *digital_output5*. If the property is assigned the value 0, the corresponding digital output turns OFF or when the value 1 is assigned, the output turns ON.
relay0 .. 3	Property (rw)	The 4 relays on the *PiXtend* board can be read as well as written to via the properties *relay0* to *relay3*. If the property is assigned the value 0, the corresponding relay turns OFF or when the value 1 is assigned, the relay turns ON.
uc_board_version	Property (r)	The *PiXtend* Board version can be read from this *read-only* property. The value 12 denotes board version 1.2.x, 13 for board version 1.3.x etc.
uc_fw_version	Property (r)	Using this *read-only* property, the firmware version of the micro-controller on the *PiXtend* board can be read.
uc_status	Property (r)	The *uc_status* property provides the current state of the micro-controller on the *PiXtend* board. A 0 means OFF or *Auto mode* is not active. In contrast, a 1 indicates that the MC is operating in *Auto mode* and has the status *Run*.
ON	Constant	Corresponds to decimal value 1.
OFF	Constant	Corresponds to decimal value 0.

6. Frequently Asked Questions (FAQ)

I cannot start my program, *Python* displays an error message about an invalid *indentation*?

There is probably a problem with the indentation of the different code lines in the program. Look again thoroughly that each line has the correct indentation. We use e.g. always 4 blanks per indentation level, if only one blank is missing, or blanks and tabs have been mixed, *Python* displays an error message. In Notepad ++ e.g. you can have the "non-printable" characters displayed, so you can see very fast where something is wrong, furthermore *Python* says itself in which line it has detected the error.

Can I run my program without *sudo*?

This seems to be possible with *Python* in conjunction with Raspbian as shown in this App-Note, but we still recommend using *sudo*, so *Python* can also perform system functions. Otherwise, problems can occur, whose cause may not be explained very easily.

The stated 100 ms cycle time is too slow for me, can't *PiXtend* be accessed faster?

Yes! This is possible. The 100 ms is a suggestion from us, so that all functions of *PiXtend* can be used. If you do not use DHT11 / 22 sensors on the GPIO0 - GPIO3, you can communicate with the *PiXtend* in a 25 ms cycle time, i.e. four times faster. But you cannot not go faster then this.

We would like to invite you to the Qube Solutions forums for information exchange:

https://www.pixtend.de/forum/

Apéndice D

Manual de usuario del PLC M-Duino 21 I/Os

En este apéndice se presentan las especificaciones técnicas del PLC M-Duino 21 I/Os.

COMPACT PLC.

INDUSTRIAL SHIELDS

2 General Description M-DUINO FAMILY product

A compact PLC based in Open Source Hardware technology. With different Input/Outputs Units.

CONECTABLE PLC ARDUINO 24Vcc M-DUINO				
MODEL TYPE	**21 I/Os**	**42 I/Os**	**58 I/Os**	
Input Voltage	12- 24Vdc			Fuse protection (1A) Polarity protection
I max.	0,5A			
Size	101x119.5x70.1	101x119.5x94.7	101x119.5x119.3	
Clock Speed	16MHz			
Flash Memory	256KB of which 8KB used by bootlader			
SRAM	8KB			
EEPROM	4KB			
Communications	I2C[1] – Ethernet Port – USB – RS485 – RS232 -- SPI – (2x) Rx,Tx (Arduino pins)			Max232-Max485-W5100
TOTAL Input points	13	26	36	
TOTAL Output points	11	22	30	
Type of signals				
An/Dig Input 10bit (0-10Vcc)	6	12	16	0-10V Input Impedance: 39K Separated PCB ground
Digital Isolated Input (24Vcc)	7	14	20	5/12/24Vdc I min: 2/6/12 mA Galvanic ISOLATION
*** Interrupt isolated Input HS** (24Vcc)	2	4	6	5/12/24Vdc I min: 2/6/12 mA Galvanic ISOLATION
Analog Output 8bit (0-10Vcc)	3	6	8	0-10 Vdc I max: 40 mA Separated PCB ground
Digital Isolated Output (24Vcc)	8	16	22	5/12/24 Vdc I max: 0.3 A Galvanic ISOLATION Diode Protected for Relay
PWM Isolated Output 8bit (24Vcc)	3	6	8	5/12/24 Vdc I max: 0.3 A Galvanic ISOLATION Diode Protected for Relay
Expandability	I2C - 127 elements - Serial Port RS232/RS485			

[1] Pull-up resistance required (IS.ACI2C-4.7K)

4 Specifications

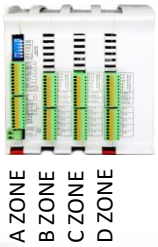
4.1 General Specifications:

Item		M-DUINO 21 IOs	M-DUINO 42 IOs	M-DUINO 58 IOs
Power supply voltage	**DC power supply**	12 - 24Vdc		
Operating voltage range	**DC power supply**	11.4 to 25.4Vdc		
Power consumption	**DC power supply**	30VAC max.		
External power supply	**Power supply voltage**	24Vdc		
	Power supply output capacity	700Ma		
Insulation resistance		20MΩ min.at 500Vdc between the AC terminals and the protective earth terminal.		
Dielectric strength		2.300 VAC at 50/60 HZ for one minute with a leakage current of 10mA max. Between all the external AC terminals and the protective earth terminal.		
Shock resistance		80m/s2 in the X, Y and Z direction 2 times each.		
Ambient temperature (operating)		0º to 45ºC		
Ambient humidity (operating)		10% to 90% (no condensation)		
Ambient environment (operating)		With no corrosive gas		
Ambient temperature (storage)		-20º to 60ºC		
Power supply holding time		2ms min.		
Weight		445g max.	542g max.	850g max.

4.2 Performance Specification:

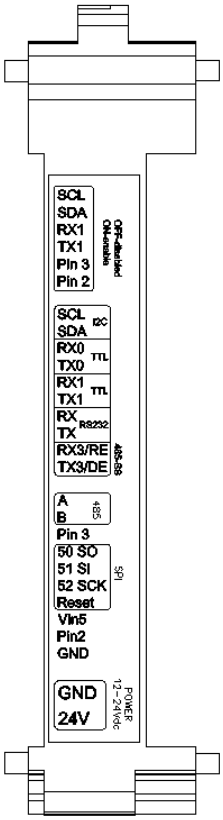
Item	M-DUINO 21 IOs	M-DUINO 42 IOs	M-DUINO 58 IOs
Arduino Board	**ARDUINO MEGA 2560**		
Control method	Stored program method		
I/O control method	Combination of the cyclic scan and immediate refresh processing methods.		
Programming language	Arduino IDE. Based on wiring (Wiring is an Open Source electronics platform composed of a programming language. “similar to the C”. http://arduino.cc/en/Tutorial/HomePage		
Microcontroller	ATmega2560		
Flash Memory	256kb of which 8 kb used by bootloader		
Program capacity (SRAM)	8kb		
EEPROM	4kb		
Clock Speed	16MHz		
Clock Speed	16MHz		

6 M-duino 21/42/58 I/O Pinout:



6.1 A Zone connection (21/42/58 I/Os)

Base (common unit)		
A Zone		
M-Duino Connector	*ArduIno Pin*	Function
SCL	*21*	I2C/SS
SDA	*20*	I2C/SS
RX0	*1*	RX0/SS
TX0	*0*	TX0/SS
RX1	*19*	RX1/SS
TX1	*18*	TX1/SS
RX	*17*	RX2(serial 2)
TX	*16*	TX2(serial 2)
RX3/RE	*15*	RX3/RS485/SS
TX3/DE	*14*	TX3/RS485/SS
A	-	RS485
B	-	RS485
PIN3	*3*	Arduino Pin/ Select SPI
50 SO	*50*	SPI
51 SI	*51*	SPI
52 SCK	*52*	SPI
Reset	*Reset*	SPI
Vin5	*Vin5*	SPI
PIN2	*2*	Arduino Pin/ Select SPI
GND	-	Gnd
GND	-	Gnd
24Vdc		

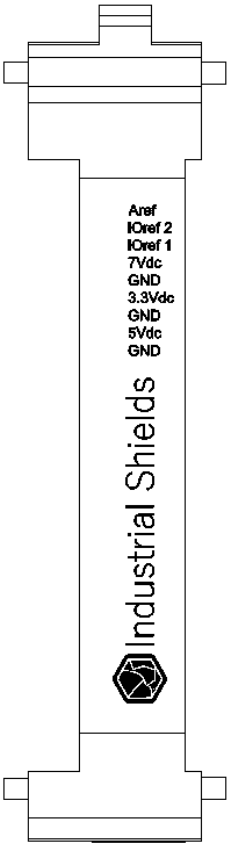


Configuration Switch* (see section 8 for Communications configuration. Enabling Communications disable s some I/Os)

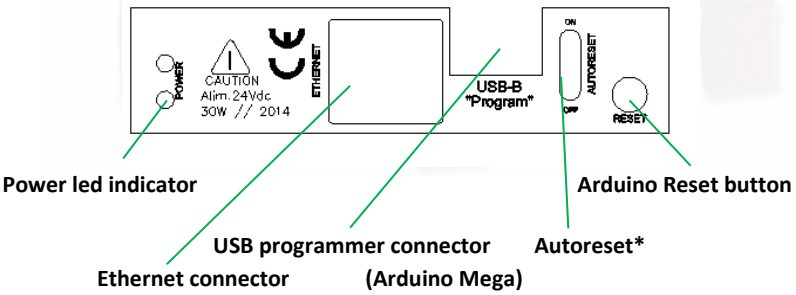
Communication Pinout

Power supply connectors (24Vdc – Gnd)

Base (common unit)		
A Zone		
M-Duino Connector	*Arduino Pin*	Function
AREF	*AREF*	Arduino PIN
IOREF2	*IOREF2*	Arduino PIN
IOREF1	*IOREF1*	Arduino PIN
7Vdc	*7Vdc*	-
Gnd	*Gnd*	GND
3.3Vdc	*3.3Vdc*	Arduino PIN
GND	*Gnd*	GND
5Vdc	*5Vdc*	-
GND	*Gnd*	GND



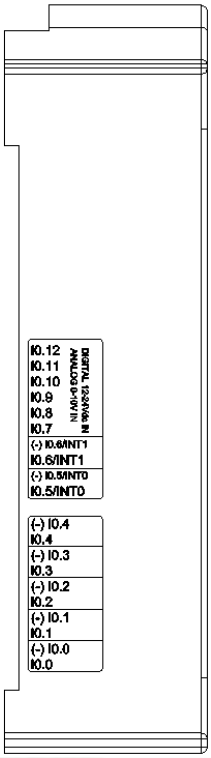
6.2 A Zone top (21/42/58 I/Os)



**NOTE: Autoreset. Arduino mega has auto reset when using serial communication code. Set switch to OFF when using serial communication. When uploading code to Arduino Mega set switch to ON.*

6.3 B Zone (21/42/58 I/Os)

B Zone		
M-Duino Connector	*Arduino Pin*	Function [2]
I0.12	*A5*	Analog/ Digital In
I0.11	*A4*	Analog/ Digital In
I0.10	*A3*	Analog/ Digital In
I0.9	*A2*	Analog/ Digital In
I0.8	*A1*	Analog/ Digital In
I0.7	*A0*	Analog/ Digital In
(-)I0.6/INT1	*NC*	GND I0.6
I0.6/INT1 [3]	*3*	Interrupt 1 In
(-)I0.5/INT0	*NC*	GND I0.5
I0.5/INT0 [3]	*2*	Interrupt 0 In
(-)I0.4	*NC*	GND I0.4
I0.4	*26*	Digital Input
(-)I0.3	*NC*	GND I0.3
I0.3	*25*	Digital Input
(-)I0.2	*NC*	GND I0.2
I0.2	*24*	Digital Input
(-)I0.1	*NC*	GND I0.1
I0.1	*23*	Digital Input
(-)I0.0	*NC*	GND I0.0
I0.0	*22*	Digital Input

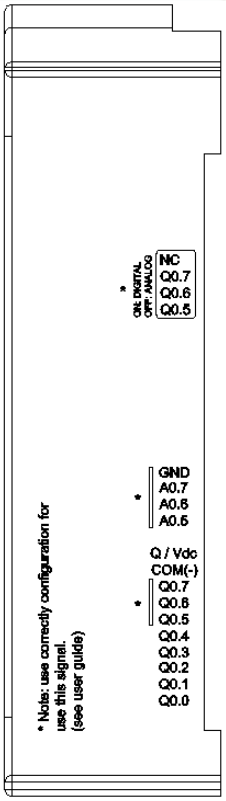


ANALOG/DIGITAL Inputs

INTERRUPT Inputs (isolated)

DIGITAL Inputs (isolated)

B Zone		
M-Duino Connector	*Arduino Pin*	Function [2]
GND	*GND*	GND
A0.7 [2]	*6*	Analog Out
A0.6 [2]	*5*	Analog Out
A0.5 [2]	*4*	Analog Out
Q/Vdc	*-*	External Isolated Out Vdc
COM(-)	*-*	External Isolated Out Gnd
Q0.7	*6*	Digital/PWM Out
Q0.6	*5*	Digital/PWM Out
Q0.5	*4*	Digital/PWM Out
Q0.4	*40*	Digital Out
Q0.3	*39*	Digital Out
Q0.2	*38*	Digital Out
Q0.1	*37*	Digital Out
Q0.0	*36*	Digital Out



Configuration Switch*
(see section 8 select correct configuration for outputs).

ANALOG Outputs

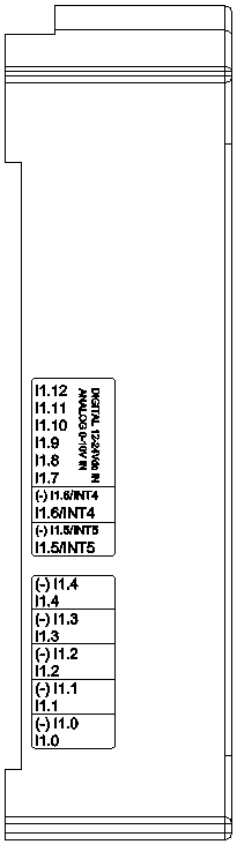
VOLTAGE SUPPLY/REFERENCE for DIGITAL/PWM Outputs (isolated)

PWM/DIGITAL Outputs

[2] See section 8 to select suitable switch configuration for (10-24Vdc/An-Dig) configurable I/Os.
[3] See section 8 to enable these connections.

6.5 C Zone (42/58 I/Os)

B Zone		
M-Duino Connector	*Arduino Pin*	Function [4]
I1.12	*A11*	Analog/ Digital In
I1.11	*A10*	Analog/ Digital In
I1.10	*A9*	Analog/ Digital In
I1.9	*A8*	Analog/ Digital In
I1.8	*A7*	Analog/ Digital In
I1.7	*A6*	Analog/ Digital In
(-)I1.6/INT4	*NC*	GND I1.6
I1.6/INT4[5]	*19*	Interrupt 4 In
(-)I1.5/INT5	*NC*	GND I1.5
I1.5/INT5[5]	*18*	Interrupt 5 In
(-)I1.4	*NC*	GND I1.4
I1.4	*31*	Digital Input
(-)I1.3	*NC*	GND I1.3
I1.3	*30*	Digital Input
(-)I1.2	*NC*	GND I1.2
I1.2	*29*	Digital Input
(-)I1.1	*NC*	GND I1.1
I1.1	*28*	Digital Input
(-)I1.0	*NC*	GND I1.0
I1.0	*27*	Digital Input

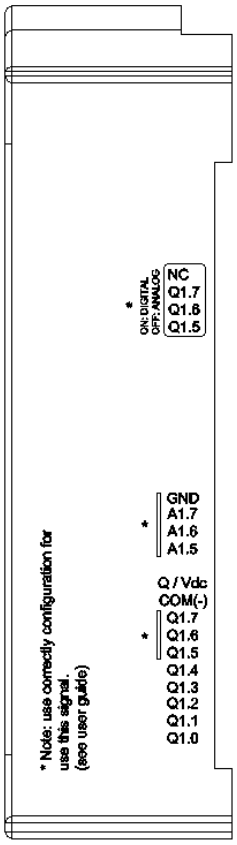


ANALOG/DIGITAL Inputs

INTERRUPT Inputs (isolated)

DIGITAL Inputs (isolated)

B Zone		
M-Duino Connector	*Arduino Pin*	Function [2]
GND	*GND*	GND
A1.7[4]	*7*	Analog Out
A1.6[4]	*9*	Analog Out
A1.5[4]	*8*	Analog Out
Q/Vdc	-	External Isolated Out Vdc
COM(-)	-	External Isolated Out Gnd
Q1.7	*7*	Digital/PWM Out
Q1.6	*9*	Digital/PWM Out
Q1.5	*8*	Digital/PWM Out
Q1.4	*45*	Digital Out
Q1.3	*44*	Digital Out
Q1.2	*43*	Digital Out
Q1.1	*42*	Digital Out
Q1.0	*41*	Digital Out



Configuration Switch*
(see section **Error! Reference source ot found.** to select correct configuration for outputs).

ANALOG Outputs

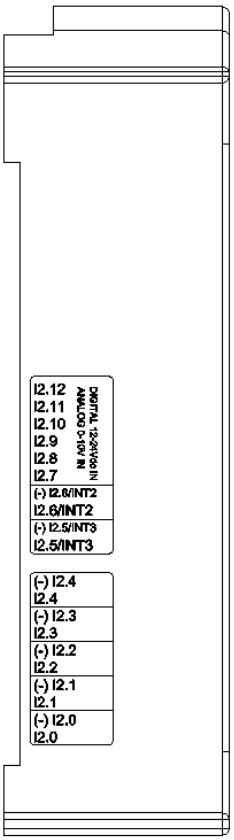
VOLTAGE SUPPLY/REFERENCE for DIGITAL/PWM Outputs (isolated)

PWM/DIGITAL Outputs

[4] See section 8 to select suitable switch configuration for (10-24Vdc/An-Dig) configurable I/Os.
[5] See section 8 to enable these connections.

6.7 D Zone (58 I/Os)

B Zone		
M-Duino Connector	*Arduino Pin*	Function [6]
I2.12	-	-
I2.11	-	-
I2.10	*A15*	Analog/ Digital In
I2.9	*A14*	Analog/ Digital In
I2.8	*A13*	Analog/ Digital In
I2.7	*A12*	Analog/ Digital In
(-)I2.6/INT2	*NC*	GND I2.6
I2.6/INT2[7]	*21*	Interrupt 2 In
(-)I2.5/INT3	*NC*	GND I2.5
I2.5/INT3[7]	*20*	Interrupt 3 In
(-)I2.4	-	-
I2.4	-	-
(-)I2.3	*NC*	GND I2.3
I2.3	*35*	Digital Input
(-)I2.2	*NC*	GND I2.2
I2.2	*34*	Digital Input
(-)I2.1	*NC*	GND I2.1
I2.1	*33*	Digital Input
(-)I2.0	*NC*	GND I2.0
I2.0	*2*	Digital Input

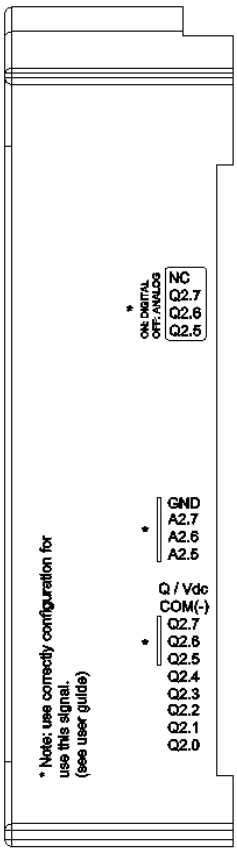


ANALOG/DIGITAL Inputs

INTERRUPT Inputs (isolated)

DIGITAL Inputs (isolated)

B Zone		
M-Duino Connector	*Arduino Pin*	Function [2]
GND	*GND*	GND
A2.7[6]	-	-
A2.6[6]	*13*	Analog Out
A2.5[6]	*12*	Analog Out
Q/Vdc	-	External Isolated Out Vdc
COM(-)	-	External Isolated Out Gnd
Q2.7	-	-
Q2.6	*13*	Digital/PWM Out
Q2.5	*12*	Digital/PWM Out
Q2.4	-	-
Q2.3	*49*	Digital Out
Q2.2	*48*	Digital Out
Q2.1	*47*	Digital Out
Q2.0	*46*	Digital Out



Configuration Switch*
(see section 8 to select correct configuration for outputs).

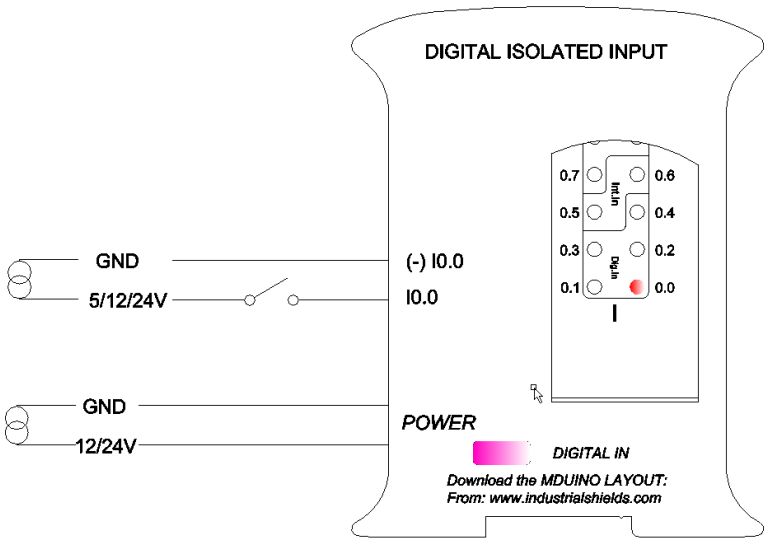
ANALOG Outputs

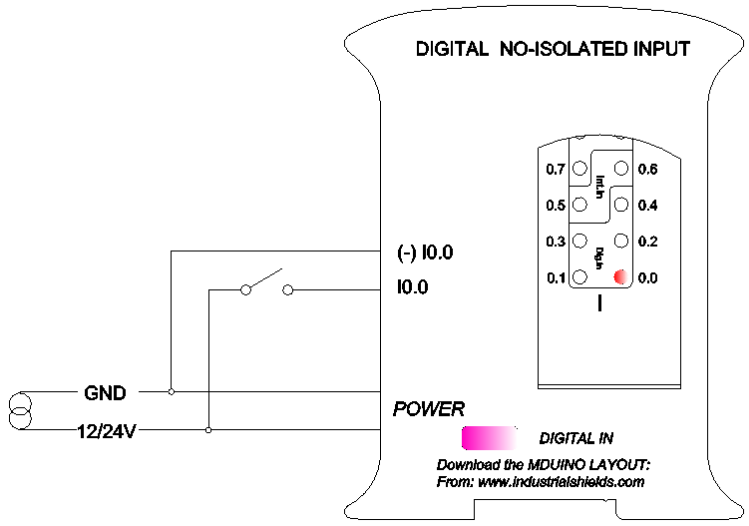
VOLTAGE SUPPLY/REFERENCE for DIGITAL/PWM Outputs (isolated)

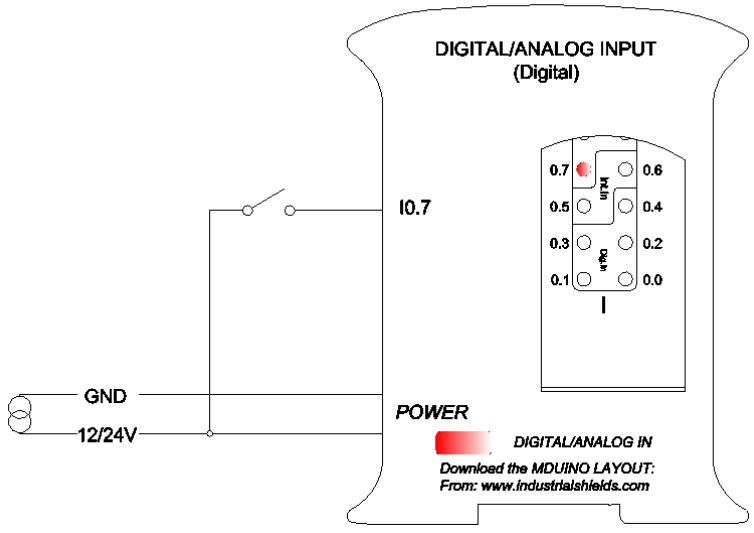
PWM/DIGITAL Outputs

[6] See section 8 to select suitable switch configuration for (10-24Vdc/An-Dig) configurable I/Os.
[7] See section 8 to enable these connections.

9. Typical Connections







Bibliografía

[1] A. Gilchrist, *Industry 4.0: the industrial internet of things*. Apress, 2016.

[2] J. A. R. Carmona, J. C. M. Benítez, y J. L. García-Gervacio, "SCADA system design: A proposal for optimizing a production line," in *Electronics, Communications and Computers (CONIELECOMP), 2016 International Conference on*. IEEE, 2016, pp. 192–197.

[3] Y. Shimanuki, "OLE for process control (OPC) for new industrial automation systems," in *Systems, Man, and Cybernetics, 1999. IEEE SMC'99 Conference Proceedings. 1999 IEEE International Conference on*, vol. 6. IEEE, 1999, pp. 1048–1050.

[4] OPC Foundation, "OPC Unified Architecture Part 1: Overview and Concepts Version 1.04," pp. 1–30, 2017. [En línea]. Disponible: https://opcfoundation.org/developer-tools/specifications-unified-architecture/part-1-overview-and-concepts

[5] S. A. Boyer, *SCADA supervisory control and data acquisition*. The Instrumentation, Systems and Automation Society, 2004.

[6] B. Galloway, G. P. Hancke *y otros*, "Introduction to industrial control networks." *IEEE Communications Surveys and Tutorials*, vol. 15, num. 2, pp. 860–880, 2013.

[7] The Modbus organization, "Modbus application protocol specification V1.1b3," pp. 1–8, 2012. [En línea]. Disponible: http://www.modbus.org/docs/Modbus_Application_Protocol_V1_1b3.pdf

[8] Z.-Y. Zhao, M. Tomizuka, y S. Isaka, "Fuzzy gain scheduling of PID controllers," *IEEE transactions on systems, man, and cybernetics*, vol. 23, num. 5, pp. 1392–1398, 1993.

[9] G. Welch y G. Bishop, "An introduction to the kalman filter. department of computer science, university of north carolina," *ed: Chapel Hill, NC, unpublished manuscript*, 2006.

[10] J. Wan, S. Tang, Z. Shu, D. Li, S. Wang, M. Imran, y A. V. Vasilakos, "Software-defined industrial internet of things in the context of industry 4.0," *IEEE Sensors Journal*, vol. 16, num. 20, pp. 7373–7380, 2016.

[11] D. F. Merchán, J. A. Peralta, A. Vazquez-Rodas, L. I. Minchala, y D. Astudillo-Salinas, "Open source SCADA system for advanced monitoring of industrial processes," in *Information Systems and Computer Science (INCISCOS), 2017 International Conference on*. IEEE, 2017, pp. 160–165.

[12] J. Glass, V. Alvarado, S. León, y J. Parra, "Politica Industrial de Ecuador 2016-2025," 2015.

[13] M. Cimoli, M. Castillo, G. Porcile, G. Stumpo *y otros*, "Políticas industriales y tecnológicas en América Latina," 2017.

[14] J. Heijs, G. M. Moles, y M. C. D. L. I. Villasol, "El impacto de las innovaciones de producto y de proceso sobre el empleo industrial."

[15] H. Lasi, P. Fettke, H.-G. Kemper, T. Feld, y M. Hoffmann, "Industry 4.0," *Business & Information Systems Engineering*, vol. 6, num. 4, pp. 239–242, 2014.

[16] M. Wollschlaeger, T. Sauter, y J. Jasperneite, "The future of industrial communication: Automation networks in the era of the internet of things and industry 4.0," *IEEE Industrial Electronics Magazine*, vol. 11, num. 1, pp. 17–27, 2017.

[17] E. Pérez-López, "Los sistemas SCADA en la automatización industrial," *Revista Tecnología en Marcha*, vol. 28, num. 4, pp. 3–14, 2015.

[18] Qube Solutions GmbH, "PiXtend V1.3 Data sheet." [En línea]. Disponible: https://www.pixtend.de/files/manuals/pixtend_v1_3_datasheet_EN.pdf

[19] Industrial Shields, "M-Duino Family." [En línea]. Disponible: https://www.robotshop.com/media/files/pdf/m-duino-plc-arduino-ethernet-58-i-os-analogdigital-plus-Datasheet-2.pdf

[20] G. Doumeingts, B. Vallespir, y D. Chen, "Methodologies for designing cim systems: A survey," *Computers in Industry*, vol. 25, num. 3, pp. 263–280, 1995.

[21] A. Gunasekaran, "Implementation of computer-integrated manufacturing: a survey of integration and adaptability issues," *International Journal of Computer Integrated Manufacturing*, vol. 10, num. 1-4, pp. 266–280, 1997.

[22] A. K. Mohamed, "Automation and computer integrated manufacturing in food processing industry: an appraisal," Ph.D. dissertation, Dublin City University, 2003.

[23] E. Gómez, L. García, y L. Hernández, "Buses de campo. estrategias de aplicación," in *Ciencias de la Ingeniería y Tecnología Handbook T-IV: Congreso Interdisciplinario de Cuerpos Académicos*. ECORFAN, 2014, pp. 291–298.

[24] T. M. S. Arik Ragowsky, "Enterprise resource planning," *Journal of Management Information Systems*, vol. 19, num. 1, pp. 11–15, 2002.

[25] L. Zheng y H. Nakagawa, "OPC (OLE for process control) specification and its developments," in *SICE 2002. Proceedings of the 41st SICE Annual Conference*, vol. 2. IEEE, 2002, pp. 917–920.

[26] M. Schwarz y J. Boercsoek, "A survey on OLE for process control (OPC)," in *Proceedings of the 7th Conference on 7th WSEAS International Conference on Applied Computer Science-Volume 7*. World Scientific and Engineering Academy and Society (WSEAS), 2007, pp. 186–191.

[27] "AMQP." [En línea]. Disponible: https://www.amqp.org/

[28] "MQTT." [En línea]. Disponible: https://mqtt.org/

[29] L. Martínez Corbillón, "OPC-UA: Un estándar al servicio de la integración en el control de edificios," *Universidad, Ciencia y Tecnología*, vol. 17, num. 66, pp. 49–53, 2013.

[30] A. Gilchrist, *Industry 4.0: the industrial internet of things*. Apress, 2016.

[31] D. Bailey y E. Wright, *Practical SCADA for industry*. Elsevier, 2003.

[32] A. Daneels y W. Salter, "What is SCADA?" 1999.

[33] H. Geng, *Internet of things and data analytics handbook*. Wiley, 2017.

[34] T. Sauter, "The three generations of field-level networks—evolution and compatibility issues," *IEEE Transactions on Industrial Electronics*, vol. 57, num. 11, pp. 3585–3595, 2010.

[35] "PROFIBUS." [En línea]. Disponible: https://www.profibus.com/

[36] "The Modbus Organization." [En línea]. Disponible: http://www.modbus.org/

[37] The Modbus organization, "Modicon Modbus protocol reference guide," pp. 1–7, 1996. [En línea]. Disponible: http://www.modbus.org/docs/PI_MBUS_300.pdf

[38] "CAN in Automation (CiA)." [En línea]. Disponible: https://www.can-cia.org/

[39] B. M. Wilamowski y J. D. Irwin, *Industrial communication systems*. CRC Press, 2016.

[40] ODVA, Inc, "EtherNet/IP." [En línea]. Disponible: https://www.odva.org/Technology-Standards/EtherNet-IP/Overview

[41] "PROFINET." [En línea]. Disponible: https://www.profibus.com/technology/profinet/

[42] "Ethernet Powerlink." [En línea]. Disponible: https://www.ethernet-powerlink.org/

[43] M. A. Johnson y M. H. Moradi, *PID control: new identification and design methods*. Springer Science & Business Media, 2006.

[44] N. Suresh, E. Balaji, K. J. Anto, y J. Jenith, "Raspberry pi based liquid flow monitoring and control," *International Journal of Research in Engineering and Technology*, vol. 3, num. 07, pp. 122–125, 2014.

[45] J. E. S. Pinzón, "Diseño e implementación de sistemas SCADA para automatismos, basados en hardware y software libre," Ph.D. dissertation, Universidad Tecnológica de Pereira. Facultad de Ingenierías Eléctrica, Electrónica, Física y Ciencias de la Computación. Ingeniería Eléctrica., 2015.

[46] P. Kadera y P. Novák, "Performance modeling extension of directory facilitator for enhancing communication in fipa-compliant multiagent systems," *IEEE Transactions on Industrial Informatics*, vol. 13, num. 2, pp. 688–695, 2017.

[47] M. Hoffmann, P. Thomas, D. Schütz, B. Vogel-Heuser, T. Meisen, y S. Jeschke, "Semantic integration of multi-agent systems using an OPC UA information modeling approach," in *Industrial Informatics (INDIN), 2016 IEEE 14th International Conference on.* IEEE, 2016, pp. 744–747.

[48] B. S. Krishna, J. Oviya, S. Gowri, y M. Varshini, "Cloud robotics in industry using raspberry pi," in *Science Technology Engineering and Management (ICONSTEM), Second International Conference on.* IEEE, 2016, pp. 543–547.

[49] R. Szabó y A. Gontean, "Industrial robotic automation with raspberry pi using image processing," in *Applied Electronics (AE), 2016 International Conference on.* IEEE, 2016, pp. 265–268.

[50] A. Girbea, C. Suciu, S. Nechifor, y F. Sisak, "Design and implementation of a service-oriented architecture for the optimization of industrial applications," *IEEE Transactions on Industrial Informatics*, vol. 10, num. 1, pp. 185–196, 2014.

[51] L. I. Minchala, S. Ochoa, E. Velecela, D. F. Astudillo, y J. Gonzalez, "An open source SCADA system to implement advanced computer integrated manufacturing," *IEEE Latin America Transactions*, vol. 14, num. 12, pp. 4657–4662, 2016.

[52] Unified Automation GmbH, "UaExpert—A Full-Featured OPC UA Client." [En línea]. Disponible: https://www.unified-automation.com/products/development-tools/uaexpert.html

[53] T. Sauter y M. Lobashov, "How to access factory floor information using internet technologies and gateways," *IEEE Transactions on Industrial Informatics*, vol. 7, num. 4, pp. 699–712, 2011.

[54] E. Molina y E. Jacob, "Software-defined networking in cyber-physical systems: A survey," *Computers & Electrical Engineering*, vol. 66, pp. 407–419, 2018.

[55] A. Creus, *Instumentación Industrial.* Alfaomega, 2010.

[56] D. F. Merchán Piedra y J. A. Peralta Sarmiento, "Diseño e implementación de un control tolerante a fallos activo utilizando plataformas de bajo costo para control, comunicación y supervisión de un proceso mediante programas de código libre y controladores lógicos programables," B.S. thesis, 2017.

[57] K. Ogata, *Ingeniería de control moderna 5 ed.* Pearson Educación, 2010.

[58] B. W. Bequette y W. B. Bequette, "Process dynamics: modeling, analysis, and simulation," 1998.

[59] L. I. Minchala-Avila, K. Palacio-Baus, J. P. Ortiz, J. D. Valladolid, y J. Ortega, "Comparison of the performance and energy consumption index of model-based controllers," in *Ecuador Technical Chapters Meeting (ETCM), IEEE.* IEEE, 2016, pp. 1–6.